PIE Writer:
Word Processing System

Program by:
SOFTWEST

HAYDEN
50 Essex Street, Rochelle Park, N.J. 07662

PIE Writer software was developed by
Thomas Crosley of SOFTWEST, Sunnyvale, California.

Copyright © 1982, Hayden Book Company, Inc.

The name PIE is a trademark of SOFTWEST. Apple, Apple II, Apple II Plus, and Applesoft are trademarks of the Apple Computer, Inc.

Apple Computer, Inc., makes no warranties, either express or implied, regarding the enclosed computer software package, its merchantability, or its fitness for any particular purpose.

FPBASIC Copyright 1977, Microsoft.
Applesoft Copyright 1978, Apple Computer, Inc.

Contents

HOW TO USE THE TEXT EDITOR (cont.)

3

The FORMAT Text Processor 29

4
PIE and FORMAT Configure 61

5
Reference Section 79

6

Advanced Topics 139

PIE WRITER: CHANGES FROM VERSION 2.0

Apple PIE Version 2.0 users will be able to use PIE Writer right away: PIE Writer keeps virtually all of the Version 2.0 features intact, while adding significant enhancements. To get the most out of your new copy of PIE Writer, take note of the following changes. For further information on new features, see the appropriate sections of this manual.

PIE Command Processor

All files created by Apple PIE 2.0 can be read and edited by PIE Writer, after converting them to 16 sector files using the Apple-supplied MUFFIN program.

The most obvious change to the Command Processor is that the E command is now used to edit an existing file, while an N command is used to create a new file. The new E command optionally allows a starting line number (with $ used to mean the last line number in the file). The old E and R commands were changed to avoid ambiguity between edit and re-edit.

The Command Processor now displays the remembered filename, length and memory left after saving and loading files, and when re-entering the CP. This information is also available using the modified LE command. It's the primary filename that is remembered: saving and loading "side files" using >>, << or line number range (begin, end) does not affect the remembered filename.

The CATALOG command has been shortened to C to save you keystrokes. Slot, drive and volume may still be optionally specified.

Whenever the program disk is accessed (when, for example, FORMAT is loaded or any CP auxiliary command is issued or you return to PIE from FORMAT) the default drive is automatically restored afterwards.

New CP commands include CAL machine language routine (which replaces the %U user function) and a CTRL-Y exit and return to and from the Apple monitor. Several new auxiliary commands (%) were added: %C outputs not only a catalog but also the number of sectors left on a disk; %FS displays a complete file status, including word and line counts; %TS and %TL save and load the PIE tab settings; %B executes a BACKUP shell to backup the current file; %H enables the help screen (it loads it into memory): and %LF toggles the line feed after return flag (for switching between peripheral devices set up in two different ways).

The quit (Q) command is now %Q, and a %M command was added to return to the main menu. The CI and CN commands are now invoked by %CI and %CN respectively. Three new auxiliary commands (%ST, %SP, and %RS) were added to interface with the DoubleTime Printer package. The cassette load and save commands (%LO and %SA), however, have been removed.

To minimize the impact on Version 2.0 shell files, the command (which is no longer documented) can still be used to display the current filename, and is identical to the modified 7 command in the current version. The file management commands (%D, %L, %U, %V, %R, and %E) no longer require (but still allows) a CTRL-D after the %.

An improved printer interface is now available from the CP: options include a printer initialization string, line feed after carriage return option, and high bit set or reset.

PIE Editor

Many of the labels for the keys have been changed (e.g. POP is now RECALL) but the key assignments all remain the same, with the single exception that CTRL-SHIFT-N now removes one word instead of one line (the latter can still be performed using ESC CTRL-SHIFT-N). On the system status line, the names PPMODE and AIMODE were changed to PPWRAP and INDENT, respectively. INDENT mode is now entered using ESC I RETURN, but ESC A RETURN has also been retained for compatibility with Version 2.0. ESC H RETURN is used to display the help screen.

Word tabbing and word delete are two major additions to the PIE editor: word tabbing (using CTRL-A and CTRL-G) is enabled by default in PPWRAP, but may be toggled between word tabbing and column tabbing in any of the three editor modes using ESC W RETURN. (Each editor mode remembers the type of tabbing you set for it). Tab settings are cleared from the top border when word tabbing is in effect. Word delete (CTRL-SHIFT-N) is available in any mode. Deletion up to a specified character is available using ESC c CTRL-J.

The only change to an existing command is that typing CTRL-S in column 1 now wraps to the last character on the previous line, instead of the last column.

Two undocumented Version 2.0 features are now fully documented in this manual: using wildcards in search strings (CTRL-Z and CTRL-Q), and entering *any* control character into a file using CTRL-SHIFT-M.

Moving or removing lines using line number (ESC # mm, nn CTRL-O or CTRL-SHIFT-N) now requires the program disk: the command must be loaded from the PIE AUX file.

The PIE editor now has a 31-character type-ahead buffer, allowing text or commands to be typed even when the system status line displays BUSY.

When the editor is ALMOST OUT OF MEMORY, the bell is rung every time a character is typed (except on the ARG line) to minimize the possibility of creating files too large for the text buffer, and worse, ultimately overwriting DDS.

After exiting PIE and then re-entering again (using the E command in the CP), PIE will return to the last line number and column except when a line number is specified after the E command. This is particularly useful in conjunction with the backup (%B) command: you can go to the CP, make a backup to disk, and with just two keystrokes, return to where you left off.

FORMAT Text Processor

Any files created by Apple PIE 2.0 can be read and formatted by FORMAT, after converting them to 16 sector files. All Version 2.0 FORMAT commands are supported by the new version, so existing files should format the same as before.

The "remembered filename" and tab settings are now preserved when switching from PIE to FORMAT and back. FORMAT can now input either binary or text files, as either the main text file or a data file. There is no longer any limit to the size of a data file. Input from cassette tape is no longer supported.

The formatted text can be output to a text file on disk, in addition to being output to the printer or screen. When text is formatted to a disk file, line feed after return is output as an option. The output representation of the new stop character escape (]+) can also be specified.

Paging to the screen is now done 24 screen lines at a time, instead of being based on the page boundaries in the document. Page boundaries are now displayed as a series of dashes when output to the screen. Page numbers can go as high as 9999.

The page offset is no longer output when formatting to the screen. When text is output to the printer or disk, the "initial offset character" is only output if it is not a space.

Several new commands were added: .tl generates a title anywhere on the page, .bf bold-faces the next input line, .cp capitalizes the next input line, .tm displays a message on the screen, .ng is used to synchronize form letter data file blocks, and .* allows insertion of comments.

The .cn, .nc, and .sc commands are included for compatability with Version 2.0 files but are not documented in this manual. The .gl command now gets lower case input, using the PIE CASE key to switch to upper case (or if installed, the shift key mode is used).

Two new escapes have been added:]+ stops on a character and waits, and]″ sends a "zero-width" control string to the printer which is ignored by the justification routines. Except for the backspace character, which has a width of −1, all control characters also have zero width.

Underlining can now be performed three ways: using backspaces, start/stop characters, or overlay. The latter two modes were added to support the Centronics 737/739 and Epson MX-80 printers. Boldface can be done in two ways (either overstriking using backspaces or using start/stop characters).

A printer initialization string (up to 7 characters) can now be sent to the printer at the beginning of a document. The default printer slot has been

changed to slot 1. And PIE Writer offers incremental spacing for "daisy-wheel" printers, such as the Diablo, Qume, and NEC.

On the RETURN TO PIE/RE-RERUN FORMAT prompts, %Q and %M and CTRL-Y function the same as in the PIE CP.

Other Changes

The CTRL-J was removed from the end of the "hello" program PIE WRITER: WORD PROCESSING and two null filenames added to the disk catalog to separate the PIE Writer files.

The SYSGEN PIE and SYSGEN FORMAT programs have been renamed PIE CONFIGURE and FORMAT CONFIGURE. Both require APPLESOFT BASIC in ROM or on the language card. But PIE and FORMAT can still be run without Applesoft.

PIE Writer now requires a 48K Apple II Plus or Apple II. Both PIE and FORMAT now start at $803 (2051) to avoid being clobbered by Applesoft. A warm start at $806 (2054) was also added. The RESET key vector for the autostart ROM is now initialized. For users without the autostart ROM, the old PIE reset vector at $803 is now located at $809.

The beginning and ending buffer pointers were moved to locations $6-$9 to stay clear of both Applesoft and Integer BASIC. Existing default variables in the $800 page were all moved up by 16 bytes ($10). However, the text and screen buffer locations were *not* changed: the text buffer in the 40-column version is still located at $4000-$9FFF, and in the 80-column versions at $4000-$9FFF. The screen buffer starts at $3300 in both versions. The PIE command tables are also still in the same location, but of course the contents have changed.

The lower part of page 3 of memory is no longer used by PIE Writer, so locations $300-$3CF are available for a user printer driver or other utilities. Page 0 locations $F8-$FF have also been reserved for user variables when calling these routines from PIE or FORMAT.

1

Introduction to PIE Writer

The PIE Writer system turns your Apple II into an easy-to-use word processor that can produce reports, letters, memos, and other business documents quickly and efficiently. As you type at the Apple keyboard, PIE Writer displays what you write on a video screen. You can easily revise the content and organization of your document while viewing it on the screen. You can add, change, or delete words, phrases, sentences, and entire paragraphs.

The PIE Writer system remembers everything you type and files it away for future use. This lets you recall a document prepared earlier, update it while viewing it on the screen, and file away the new version.

Finally, the PIE Writer system lets you enter codes, along with your text, that command the computer to set indents, insert blank lines, center heads, begin new pages, and right-justify paragraphs. These codes appear only on your video screen, not on your printed report. You can type sentence after sentence without concern for the document's final appearance, then review the entire report, edit it, enter layout codes, and print out as many copies of the perfect, final document as you want.

The PIE Writer system consists of a number of computer programs, all working together and contained on one diskette. For you, the "user," there are only three programs you need to get started: the PIE Text Editor and Command Processor, and the FORMAT Text Processor. The Text Editor lets you enter and edit text so that you can write error-free letters and reports. The Command Processor performs all the PIE Writer "housekeeping" functions, such as getting a file from the diskette into the computer so you can work with it and filing away the changes you make. The Text Processor controls how your document will look when it is printed.

HOW TO USE THIS MANUAL

This documentation has been organized so that both the beginner and the seasoned professional can use it.

The first three sections of the manual provide all the information about PIE Writer you need to start producing perfect letters and reports.

Section 1 starts at the very beginning: how to get the system started, from turning on the Apple to handling diskettes. The introduction progresses through such basic word processing concepts as what the "cursor" is and how it works, how to view text in the "window," and what "scrolling" means.

Sections 2 and 3 introduce you, in order, to the Text Editor, the Command Processor, and the Text Processor.

In the last three sections, PIE Writer's truly outstanding features are discussed in greater, more technical, detail.

Section 4 begins the more advanced material, intended for users who have mastered the basic features of PIE Writer. Here, you learn how to adjust the PIE Writer system to work with any Apple II and any printer, and how to modify the system to provide for specific, "permanent" document layouts.

Section 5 reviews all the basic PIE and FORMAT commands already covered and introduces additional, more powerful, commands that can be brought into use as you gain experience. A summary of all system error messages explains how you can recover from any mistake.

Section 6 describes advanced PIE Writer features of interest to word processing professionals and programmers. The experienced user learns how to customize form letters through "file switching," how to transmit over telephone lines from the Apple to another computer, and even how to have custom-made software work with PIE Writer. Address tables are included.

The index gives you quick and easy access to all PIE Writer terms and topics.

Finally, a Reference Card lists all the PIE Writer commands, their meanings, and uses. The card also has a chart of suggested key labels, single words that serve as memory aids in learning the simple commands. You can use this handy card whenever you are "stuck" for the right command.

BASIC CONCEPTS

Before you start doing any word processing, there are a few key words and concepts you should know. It is important that you read through this section carefully and understand these concepts before you turn on your system.

The Cursor

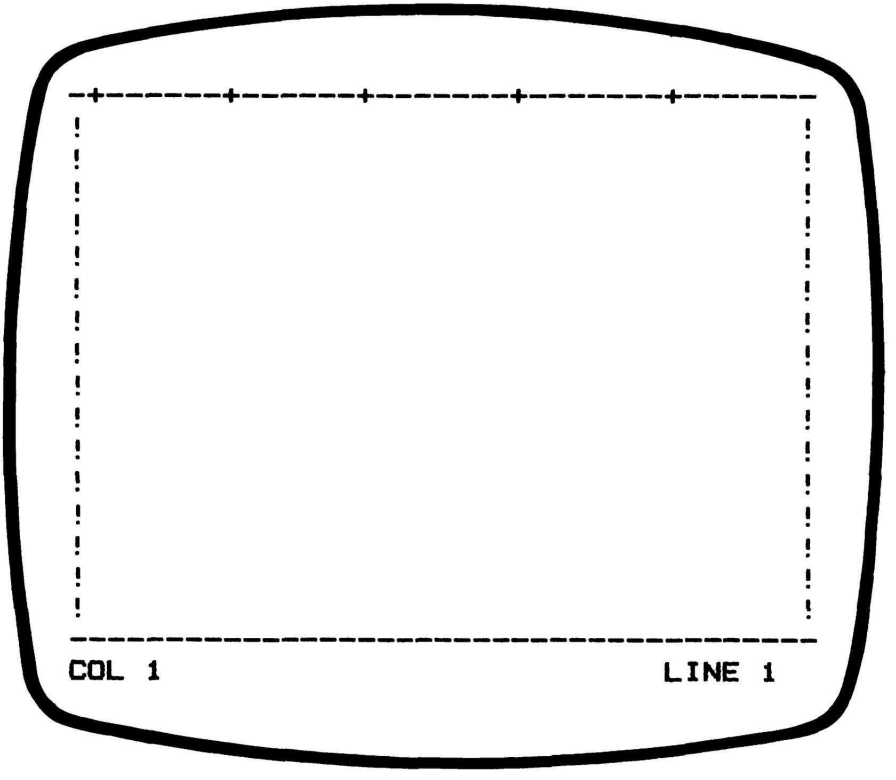
When you first use PIE Writer, a blinking rectangle, the cursor, will appear on your screen to the right of the phrase "WHICH ONE?". The cursor indicates your typing position on the screen, showing you exactly where your next typed character will appear.

When you get into the Text Editor program, the cursor will flash in the top left-hand corner of your screen. You can enter text from the keyboard that will appear on the screen at the cursor, or you may move the cursor to any other position on the screen and begin entering text at that point.

You will learn how to move the cursor around the screen in section 2.

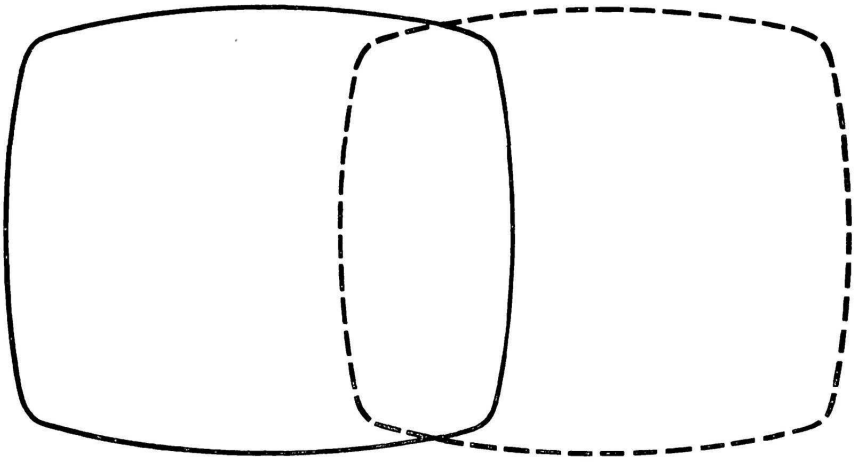
The Window

With the PIE Writer system, your video screen becomes a "window" through which you can see the text you have entered from the Apple keyboard. The window can display 38 characters across the screen and 21 lines of characters down the screen. If your Apple has an 80-column board in it, the window will display 78 characters across and 21 lines down. {NOTE: From now on, all references to 80-column systems will be in brackets.}



Beyond the Right-Hand Margin

Although the window can show only 38 {or 78} columns across at a time, you can still type in documents up to 64 {or 128} characters wide on the screen. The reason for this is that the window will shift automatically to the right as you type the 39th {or 79th} character in a line, as shown below:



The right-hand side of the "page" reveals extra typing room.

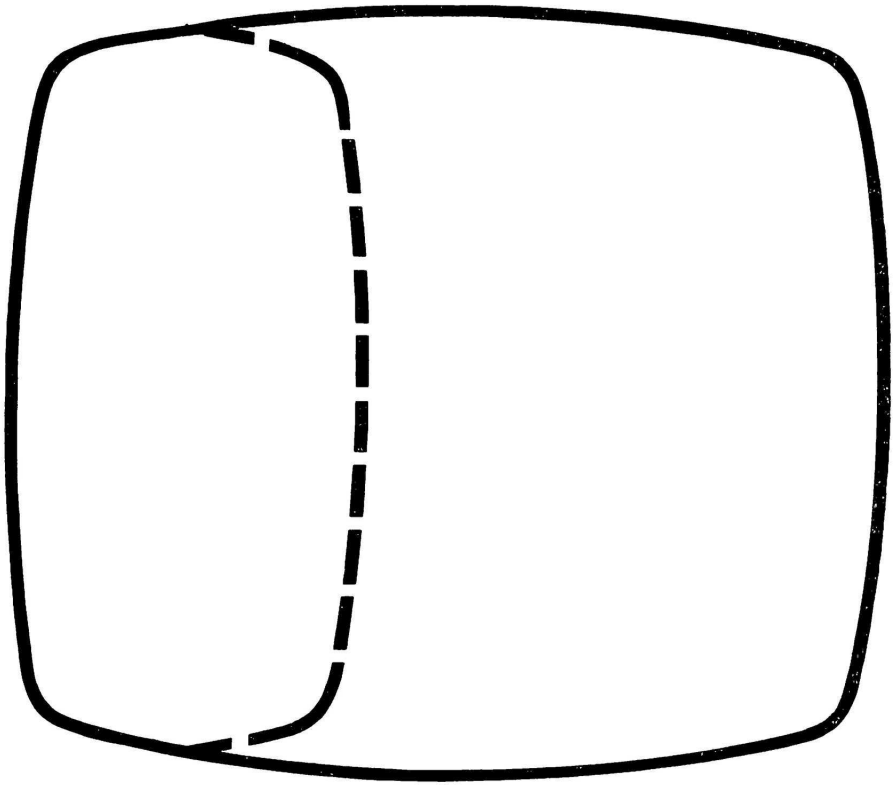
When the window shifts to the right, you may feel lost for a moment because the right-hand side of the page partially overlaps the left-hand side. Characters 27 to 38 {or 51 to 78} from the left-hand side are displayed at the beginning of the line as shown in the diagram to the right.

One last point about what you type and the confines of the window: No matter how wide or narrow you choose to type your lines on the screen, you can direct the PIE Writer system to print your document any width you desire. The window only restricts the number of characters you can see on the screen at once, not the final appearance of the document.

Scrolling

Most of the documents you type are likely to exceed 21 lines in depth. PIE Writer lets you type a document of any length.

Just as the window is capable of showing only 38 {or 78} characters across at a time, it can show only 21 lines of text at a time. The page showing in the window can be "scrolled" backward, toward the beginning of your document, or forward, toward the end of the document. This lets you view the entire document on the screen before you print it out.



It is important to remember that in order to see text above or before your current cursor position, you must scroll the page "down." To review material that follows your current cursor position, you must scroll the page "up."

Commands

There are three types of commands: single, double, and sequential. All the commands you type are shown in bold throughout the manual.

- *SINGLE KEY COMMANDS require that you press only one key in order to have the computer perform a function. Some single key commands will execute the function immediately once the key is pressed. Most single key functions, though, require that you press* **RETURN** *after entering the command before the computer will execute it.*

- *DOUBLE (or TRIPLE) KEY COMMANDS consist of two (or three) keys that must be held down at the same time. These commands are executed immediately, so the* **RETURN** *is not needed. The convention used to indicate a double key command is a hyphen between the keys, as in* **CTRL-A**.
- *SEQUENTIAL COMMANDS use more than one key but do not require that the keys be held down together. The key shown first is typed first; the key shown second is typed second; and so on. Sequential commands must be followed by a* **RETURN**. *In this manual, sequential commands are shown in the order they should be typed, without hyphens, as in* **ESC W RETURN**.

Single	Double	Sequential
E, then **RETURN**	**CTRL-A**	**ESC W RETURN**

Default Values

Default values are a series of specifications built into the PIE Writer system. The specifications represent values that most people would use in composing text. For example, if you tell PIE Writer that you want your paragraphs indented, but do not tell it how many spaces, it will indent 5 spaces automatically. If you want, you can "override" the default value by telling PIE Writer exactly how many spaces you want it to indent each paragraph.

In section 4, you will learn how to set *new* PIE Writer default values to suit your own particular needs.

Final Suggestions

With these few basic concepts, you are now ready to begin using PIE Writer. Some final suggestions before you start:

1. *Do not be too concerned about making mistakes. PIE Writer can handle an infinite number of typographical and spelling errors. If you follow the instructions provided for handling your diskettes correctly (on the back of your PIE Writer diskette protective paper envelope) and insert them in the disk drive properly, you should have no mechanical problems with your diskettes, your disk drive, or your Apple computer.*
2. *Keep your PIE Writer manual handy at all times.*

NECESSARY EQUIPMENT

To use PIE Writer, you will need:

- *an Apple II, with Applesoft, or an Apple II Plus computer with 48K of memory*
- *the PIE Writer diskette*
- *a disk drive and interface card*
- *a television set or monitor*
- *a printer and interface card*
- *extra blank diskettes*

The PIE Writer system assumes that you have all the equipment listed above. It also assumes, of course, that your equipment is installed correctly. It is important to note that:

- *The printer interface card should be in slot 1.*
- *The disk drive interface card should be in slot 6.*
- *The 80-column board, if you have one, should be in slot 3.*

If your system is not installed this way, see section 4 on reconfiguring PIE Writer to work with your hardware.

PIE Writer is versatile, so it can work with almost any combination of printers, lowercase adapters, and 80-column display boards. If you have any difficulty installing your hardware, refer to Apple II reference manuals for information on the Apple's Disk Operating System or consult your local dealer.

GETTING STARTED

The first step in using the PIE Writer system is to learn how to "boot" your PIE Writer system diskette. This loads the first PIE Writer program file into the computer.

To begin:

1. *Turn the Apple computer switch, located at the left rear of the computer, to OFF.*

2. *Take the PIE Writer system diskette in your hand (label facing upward, write-protect notch to the left, oval-shaped gap facing away from you)*

3. *Insert the diskette gently into your disk drive. The diskette will move the last fraction of an inch by itself, properly seating the diskette. Make sure that you insert your system diskette in drive 1 if you have more than one disk drive, or it will not boot.*

4. *Close the door of the drive.*

5. *Turn ON the Apple computer, again using the switch at the left rear of the computer.*

The disk drive will whirr and click, and the disk drive's red IN USE light will glow. The PIE Writer diskette is now loading itself.

If Additional Help Is Needed

If you should encounter any difficulty booting your system diskette, the problem may be solved by doing one of the following:

1. *If you do not have Autostart ROM, a feature in all but the oldest Apple computers that allows you to boot a diskette more easily, you will see an asterisk (*) on your video screen. This is a "prompt," meaning that the computer is waiting for you to give it additional instructions. When you see the *, type* **6 CTRL-P** *then press* **RETURN**, *and the diskette should boot.*

2. *If it still does not boot, open your Apple computer and make sure that your disk interface card is in slot 6 and that the disk drives are hooked up to the card correctly. Turn off the computer, make the necessary adjustments, and turn on the computer again. Type* **6 CTRL-P** *then press* **RETURN** *in response to the * prompt. If your diskette still does not boot, consult your dealer.*

3. *Make sure to follow the instructions on the back of your PIE Writer diskette protective paper jacket for taking care of your PIE Writer diskette. Mishandling can result in loss of information or permanent damage of the diskette.*

Backing Up Your PIE Writer Diskette

IMPORTANT: If this is the first time you have booted your system diskette, you should make a backup copy of it IMMEDIATELY. Select **5** in response

to the System Menu's WHICH ONE? prompt, then make a copy of your disk as instructed on p. 38 of the Apple II DOS Manual, supplied by your dealer.

Place your original PIE Writer diskette in a safe place, and use your backup as your working system diskette.

The System Menu

When your PIE Writer diskette boots successfully, the System Menu, as shown below, will appear on your video screen:

```
PIE WRITER: WORD PROCESSING SYSTEM
     (C) 1982 HAYDEN BOOK COMPANY

              SYSTEM MENU


          1 - PIE TEXT EDITOR
          2 - FORMAT TEXT PROCESSOR
          3 - PIE CONFIGURE
          4 - FORMAT CONFIGURE
          5 - EXIT

              WHICH ONE? □
```

A NOTE TO 80-COLUMN USERS: The first time you boot PIE Writer, instead of seeing the System Menu, you will be in the 80-column configuration program. This configures PIE Writer to work with your specific 80-column board. Each 80-column board requires a slightly different routine to display characters on the screen. The 80-column configure program "patches" PIE Writer with the appropriate video output routine.

IT IS CRITICALLY IMPORTANT that you make a backup copy of the original PIE Writer diskette before working with the 80-column configure program. The 80-column process deletes files that contain the video drivers for the other 80-column boards. The original diskette should be stored safely, and the configuration program run (after booting the diskette) using the backup copy.

Assuming you are working with a copy of the system diskette, the first step is to confirm that you have made a backup when prompted by the 80-column configure program.

Next, a list of the 80-column boards your version of PIE Writer supports is displayed. You are asked either to choose one of them or to exit the program.

Finally, you are asked to confirm your selection. Once you have done this, PIE Writer begins the configuration. This takes a few minutes, so be patient.

When the configuration is complete, the System Menu described above is displayed. The PIE Writer system is now configured to run with your 80-column board.}

Now you are ready to begin working with the Text Editor.

2

The PIE Text Editor and Command Processor

PIE and FORMAT work together in the PIE Writer system. FORMAT allows you to layout and print a document according to your specifications (see "The FORMAT Text Processor," on p. 29). The PIE Command Processor manages files that you create with the Text Editor.

The system diskette includes a lesson that teaches you all the Text Editor commands you will need to enter and edit your text on the screen. However, you can experiment with the Text Editor on your own if you want.

HOW TO USE THE TEXT EDITOR

When you boot the system diskette, the System Menu is displayed on the screen:

```
        SYSTEM MENU

1 - PIE TEXT EDITOR
2 - FORMAT TEXT PROCESSOR
3 - PIE CONFIGURE
4 - FORMAT CONFIGURE
5 - EXIT

    WHICH ONE? □
```

1. *Type* **1** *to respond to the "WHICH ONE?" prompt. The disk drive will whirr briefly, then the next screen of information will be displayed:*

```
FILENAME:        ?

LENGTH:                    0
MEMORY LEFT:   22015

COMMAND?: [
```

2. *To take the Text Editor lesson, type* **LOAD LESSON** *and press* **RETURN**. *If you mistype, press* **CTRL-X** *to cancel the line, then repeat this step. If you would rather use the Text Editor without taking the lesson, go directly to step 3 instead. If you have loaded LESSON, the next display is:*

```
FILENAME:        LESSON

LENGTH:                    4863
MEMORY LEFT:   17152

COMMAND?: [
```

3. *Type* **%H**, *then press* **RETURN**. *This option loads the PIE HELP file into memory. When the COMMAND?: prompt is displayed again, type* **E**, *then press* **RETURN**.

If you have loaded the lesson file, the first of eighteen Text Editor lessons appears on your screen. First, read the introduction to the lesson. Note that each new Text Editor command is listed in bold with a brief description of its function. Then, follow the instructions on the screen for each lesson. (All of the lessons, as they appear on the screen, are reproduced in section 5 on p. 98.)

If you are working in the Text Editor without the lesson, practice the commands as they are introduced. Enter text anywhere on the screen. If you go beyond the right-hand margin and the window shifts to the right, press **RETURN** to get back to the main page.

To begin the whole lesson over, or to go back to where you began working with the Text Editor, type **CTRL-T**.

Finally, all of the simple Text Editor commands have been given one-word labels to help you remember the function performed when those keys are pressed. Throughout the lessons and the manual, the labels appear in parentheses after the command, as in **CTRL-D** (HOME). Do not type these as part of the commands; they are only reminders.

If you still need help remembering the function of certain commands, press **ESC H RETURN** to display all the simple Text Editor commands and their functions; press the **RETURN** key to return you to the point in the file where you left off.

Lesson 1: The Cursor

When the first Text Editor window appears on your screen, the cursor flashes in the top left-hand corner of the window.

When you want to scroll forward to the next page, use the **CTRL-V** (PAGE ⇩) command. The **CTRL-V** command advances the screen and cursor 21 lines, or 1 page, at a time.

Command Key (Label)	Simple Function
CTRL-V (PAGE ⇩)	Advances screen 21 lines (1 page) at a time

Lesson 2: Home and Bottom Home

The second lesson describes the function of the **CTRL-D** command (HOME). The **CTRL-D** is a toggle command—indicated by an asterisk under "Command Key" below—that moves the cursor to the top left-hand and the bottom left-hand corner of the screen.

Press **CTRL-D** to send the cursor Home from any cursor position on the screen. Press **CTRL-D** again to send the cursor to Bottom Home.

Command Key (Label)	Simple Function
CTRL-D (HOME)*	Moves the cursor to the top left-hand or bottom left-hand corner of the screen (toggle)

Lesson 3: Simple Cursor Movement

The simplest cursor movements (up, down, left, and right one space) have been positioned on your keyboard so that you can perform them quite easily with one hand. These commands—**CTRL-E** (⇧), **CTRL-C** (⇩), **CTRL-S** (⇦), and **CTRL-F** (⇨)—combine with the **CTRL-D** to form a lopsided cross on the keyboard.

Notice that when you move the cursor with these commands, the text the cursor passes over remains intact.

Command Key (Label)	Simple Function
CTRL-E (⇧)	Moves the cursor up 1 line
CTRL-C (⇩)	Moves the cursor down 1 line
CTRL-S (⇦)	Moves the cursor 1 space to the left
CTRL-F (⇨)	Moves the cursor 1 space to the right

Lesson 4: Cursor Movement Practice

Practice the cursor commands you have just learned. These commands are probably used more often than any other group of commands, so it is important to become familiar with them before continuing.

The **REPT** key repeats a single command as many times as you want, for as long as you hold down the **REPT** key and the command sequence at the same time. Use the **REPT** key to make the cursor move up, down, left, or right faster.

Command Key (Label)	Simple Function
REPT-	Repeats any command for as long as you hold down the **REPT** key and the command at the same time

Lesson 5: The Status Line

The Status Line is displayed just below the bottom border of the window on the screen. The Status Line displays information about the cursor position and the current text entry "mode."

The Status Line also keeps track of other details, such as * LOCASE *, and displays them on the screen. Keep an eye on the Status Line as you work.

Lesson 6: Beyond the Right-Hand Margin

The PIE Writer screen can display only 38 {or 78} characters horizontally at a time. There is additional typing room beyond the right-hand margin, though.

A right caret (>) on the right-hand side of the main page indicates that there is text beyond the right-hand margin on that line. Use the **CTRL-F** to move the cursor to the right until the right-hand side of the page appears.

As you type the 39th {or 79th} character in the line, you will hear the bell column tone. The bell column is the 38th {or 78th} column. It is displayed as a reverse video dash at the top right-hand corner of the main page window. This setting is a system default. You can set the bell tone in any column you want, or clear the bell column completely, with an advanced command (introduced in section 5).

Press **RETURN** to get back to the main page.

Lesson 7: Tabs

You can move back and forth across the page quickly by using the tab commands, **CTRL-G** (TAB ⇨) and **CTRL-A** (TAB ⇦). The **CTRL-G** moves the cursor one tab stop to the right; the **CTRL-A** moves the cursor one tab stop to the left.

Each tab stop is indicated by a plus (+) sign at the top of the window. These settings are a system default. You can change the position and the number of tab stops with more advanced tab commands (introduced in section 5).

If you tab the cursor beyond the right-hand margin, you can get back to the main page by tabbing left with the **CTRL-A** or by pressing **RETURN**.

Command Key (Label)	Simple Function
CTRL-G (TAB ⇨)	Moves the cursor 1 tab stop to the right
CTRL-A (TAB ⇦)	Moves the cursor 1 tab stop to the left

Lesson 8: Moving the Cursor to the Beginning or End of a Line

The **CTRL-B** (⇦⇨) command, like **CTRL-D**, toggles back and forth between two different functions. Press **CTRL-B** once to move the cursor to the end of the line, even if the end of the line is beyond the right-hand margin. Press **CTRL-B** again to move the cursor back to the left margin.

The **CTRL-B** command makes it easy to add text at the end of a line or begin a new sentence there.

Command Key (Label)	Simple Function
CTRL-B (⇦⇨)*	Moves the cursor to the beginning or end of a line (toggle)

Lesson 9: Scrolling Pages and Lines

You have been taught how to scroll the entire page forward using **CTRL-V** (PAGE ⇩). You also can scroll the page up or down a screen or a line at a time.

CTRL-R (PAGE ⇧) lets you scroll back a page at a time to review text.

You also can scroll up a single line at a time with **CTRL-Y** (SCROLL ⇧) and down a line at a time with **CTRL-N** (SCROLL ⇩).

Command Key (Label)	Simple Function
CTRL-R (PAGE ⇧)	Reverses screen 21 lines (1 "page") at a time
CTRL-Y (SCROLL ⇧)	Reverses screen and cursor 1 line at a time
CTRL-N (SCROLL ⇩)	Advances screen and cursor 1 line at a time

Users with lowercase adapters who have installed them (see p. 7) and have altered FORMAT Configure (see p. 72) correctly, or who have 80-column boards, can skip the next lesson because their video screens display true upper- and lowercase characters. Standard Apple systems do not.

Lesson 10: Upper- and Lowercase Character Display

The screen displays all characters in uppercase. It distinguishes between the upper- and lowercase characters you enter at the keyboard, though, by showing uppercase in reverse video (black letters on a white background) and lowercase normally (white letters on a black screen).

You enter uppercase characters with the ➡ (CASE) key, as explained below.

Lesson 11: Uppercase Text Entry

The ➡ (CASE) key lets you display uppercase letters on the screen and print them out as uppercase.

Press ➡ once to enter the next character you type in uppercase. You are returned to lowercase entry mode immediately after entering the single uppercase character.

Press ➡ twice to lock in uppercase. Then enter as many uppercase characters as you want. You are not returned to lowercase entry mode until you press ➡ twice more.

IMPORTANT: The ➡ key is not to be confused with the **CTRL-F** (⇨). Remember that the right arrow (⇨) following the **CTRL-F** is only a label meant to remind you that the **CTRL-F** command moves the cursor one space to the right. The ➡, whose label is (CASE), is a real button on the Apple keyboard.

Command Key (Label)	Simple Function
➡ (CASE)	Makes next character entered uppercase
➡ ➡	Locks in and unlocks uppercase mode (toggle)

Lesson 12: Entering Upper- and Lowercase Characters

Enter one or more letters in uppercase, in combination with lowercase letters. Type all over the screen. Practice toggling the ➡ (CASE) key back and forth to learn easy upper- and lowercase character entry.

If you shift right after typing the 39th {or 78th} character in a line, press **RETURN** to get back to the main page.

Lesson 13: Upper-Register Characters

The **SHIFT** key, not to be confused with the ➡ (CASE) key, lets the user enter upper-register symbols and punctuation marks. Hold down the **SHIFT** key while typing one of the Apple's upper-register characters, which are: ! " # $ % & ' ( ) * = @ + < > and ?. These symbols appear in plain white on a black screen, whether they are entered in upper- or lowercase mode.

If the window shifts right after you type a line's 39th character, press **RETURN** to see the main page.

Command Key (Label)	Simple Function
SHIFT	Hold down the **SHIFT** key and an upper-register character at the same time to enter that character

Lesson 14: Destructive Cursor Moves

The first thirteen Text Editor lessons cover cursor control, window control, and text entry. Lessons 14 through 17 teach simple editing commands.

The **space** bar and ◀ (BACK) key each perform a simple cursor movement, but they also erase whatever text characters are in their path.

The **space** bar deletes a character, then moves the cursor 1 space to the right of where that character was.

The ◀ (BACK) key deletes a character to the left as it backspaces.

Command Key (Label)	Simple Function
space bar	Deletes a character and moves the cursor 1 space to the right
◀ (BACK)	Deletes a character to the left as it backspaces

Lesson 15: Deleting a Character

The **CTRL-J** (DELETE) key deletes a character or a blank and shifts the rest of a line left toward the cursor. Position the cursor on any letter you want to delete, and type **CTRL-J** to edit a misspelled word.

Command Key (Label)	Simple Function
CTRL-J (DELETE)	Deletes a character at the cursor and moves the other characters on the line 1 space to the left

Lesson 16: Inserting a Character

The **CTRL-P** (INSERT) allows you to enter characters anywhere within a word or a line.

The **CTRL-P** toggles between entering and leaving the insert mode. Press **CTRL-P** to enter the insert mode, then enter as many characters in the middle of a word or a line as you want. All the characters on the line, beginning with the character at the cursor position when you first enter the insert mode, shifts right as you enter text.

Press the **CTRL-P** again to leave the insert mode.

Command Key (Label)	Simple Function
CTRL-P (INSERT)*	Begins or ends the insertion of characters at the cursor

Lesson 17: Shifting Text Left or Right

You can shift a line of text left or right while you are in the insert mode.

First, type **CTRL-P** to enter the insert mode.

Now, to shift a line to the right, press the **space** bar. All text on the line at or to the right of the cursor position before you entered the insert mode shifts right.

To shift a line to the left, press the ⬅ (BACK) key. The ⬅ key lets you "drag" a line to the left.

The editing commands taught in these last few lessons should make one thing clear: You need not enter text perfectly the first time with PIE Writer. You can always review and edit your document later.

Command Key (Label)	Simple Function
CTRL-P (INSERT) **space**	Shifts a line of text to the right 1 space
CTRL-P (INSERT) ⬅ (BACK)	Shifts a line of text to the left 1 space

Lesson 18: Returning to the Beginning of a File; Leaving the Text Editor and Entering the Command Processor

Type **CTRL-T** (GOTO) to return to the first page of your file. **CTRL-T** takes you to the beginning of any file created with the Text Editor.

Remember that you can scroll backward or forward a page or a screen at a time. To go back to a specific place in your file, use the **CTRL-R** (PAGE ⇧), **CTRL-V** (PAGE ⇩), **CTRL-Y** (SCROLL ⇧), and **CTRL-N** (SCROLL ⇩).

When you have finished working with the Text Editor, type **CTRL-SHIFT-P** (EXIT) to leave the Text Editor and enter the Command Processor.

Command Key (Label)	Simple Function
CTRL-T (GOTO)	Moves the cursor to the beginning of a file
CTRL-SHIFT-P (EXIT)	Leave the Text Editor and enter the Command Processor

The manual documents more sophisticated and powerful word processing commands that can be used by any PIE Writer user. All these commands are introduced in section 5.

When you leave the Text Editor with a **CTRL-SHIFT-P** , you enter the Command Processor.

THE COMMAND PROCESSOR

The Command Processor (CP) manages all the operating files on the system diskette and all the document files you create using the Text Editor. Some of the functions of the CP are:

1. *Loading a file from the diskette so you can review, edit, or add to it.*
2. *Saving text you have written on a diskette file and labeling the file with a name you specify.*
3. *Clearing the memory so that you can begin a brand new file.*
4. *Reminding you of the filename of the text you are working on.*
5. *Reporting on the number of words or lines in a file.*
6. *Transferring you back and forth between the CP, the System Menu, and the Text Editor.*
7. *Sending text you have written to be printed using the FORMAT Text Processor.*

Again, these are only a few of the Command Processor's capabilities.

Before you learn any commands, there is one more concept with which you should become familiar: the "text buffer."

Saving the Text Buffer

The text buffer can be thought of as the computer's short-term memory. It sets aside a certain amount of space for you to enter a document and remembers all the text that you enter. But it remembers your file only while you are working on it. To store the text that is in the buffer for future reference, you must "save" it on a diskette.

IMPORTANT NOTE: Everything in the buffer will be erased completely if any one of the following happens:

- *You turn the computer off or the power goes off.*

- *You load another file from the diskette for review or editing. The file from the diskette replaces the text that was in the text buffer.*
- *You tell the CP to begin a new file. The text buffer will clear out any text it is holding and stand by to receive new text.*

Before loading another file, beginning a new file, or whenever you have typed anything you cannot afford to lose, ALWAYS make sure you save the contents of the text buffer. Get in the habit of saving often to save yourself many hours of grief.

You will learn how to save a file later. First, you will learn how to begin a new file.

Beginning a New File

Every time you want to prepare a new document, you must begin a new file for it. To begin a new file, with the COMMAND?: prompt flashing on the screen:

COMMAND?:	1. *Type* **N** *(or* **NEW***)* 2. *Press* **RETURN**

The N you type tells the CP to clear out the text buffer and prepare to receive new text; **RETURN** signals that you have finished entering your command and are ready to have it carried out. The CP requires a **RETURN** after every command before it can perform the required function.

Before clearing the buffer, the CP will give you one last chance to save any text in the text buffer, prompting as shown below on the left. Since you have not typed anything into the buffer yet, it is safe to give the go-ahead, shown on the right:

OK TO CLEAR?:	1. *Type* **Y** *(for Yes)* 2. *Press* **RETURN**

The text buffer window appears. This is the first blank page onto which you may begin entering text.

You have begun a new file, leaving the Command Processor and entering the Text Editor for the moment.

CP Command Review	Function
N (or **NEW**) **RETURN**	Clears the text buffer, and opens a new file

Typing Modes

When you enter the Text Editor for the first time, you can begin entering text in the system's default Manual mode.

To enter text smoothly in Manual mode, you need make no adjustments, just follow the directions below:

- *When you get to the end of a line, press* **RETURN**.
- *If you type onto the right-hand side of the page, press* **RETURN** *to get back to the main page.*

After you have typed long enough to get a feel for the restrictions of Manual mode (pressing **RETURN** at the end of each line), you can switch to PPWRAP mode for easier text entry. To change the system default and use the PPWRAP mode:

1. *Press* **ESC**
2. *Press* **RETURN**

Note that when you enter PPWRAP mode, the Status Line says "PPWRAP."

In PPWRAP mode, your typing speed should almost double, because you do not need to press **RETURN** when the cursor reaches the end of a line. The lines "wrap" around the screen automatically.

PIE Writer is now set for you to start typing streams of sentences in PPWRAP mode without having to press **RETURN**.

Modifier	Command Key (Label)	Function
ESC	**RETURN**	Sets PPWRAP typing mode

When you enter the PPWRAP mode, you can tab to the beginning of words to the left (**CRTL-A**) or right (**CRTL-G**) of the cursor position. Notice that the plus (+) signs disappear from the top of the screen.

Again, there are many additional cursor, screen control, and editing commands available in the Text Editor. These commands, providing such features as word tabbing, searching and replacing, and line moving are covered in section 5.

Now, though, you will learn how to get back to the Command Processor and save a file. When you have practiced enough typing in the PPWRAP mode, read on.

Saving a File

Right now, everything you have typed exists only in the text buffer. To place your text in the computer's long-term memory, you must save a copy of it on the diskette.

The first step in saving a file is to leave the Text Editor and enter the Command Processor with a **CTRL-SHIFT-P** (EXIT) command.

After you leave the Text Editor, your text will disappear and you will be prompted by the CP:

```
COMMAND?:
```

To save the file:

1. *Type* **S** *(or* **SAVE***).*
2. *Type at least one blank space.*
3. *Type a filename of your choice. (For users with 2 disk drives, you may type* **,D2** *after the filename.) Filenames must begin with a letter; they cannot contain any commas; and they must be no more than 30 characters long, including blank spaces. If you do not begin your filename with a letter, or if your filename contains a comma, you will receive a PIE command error message:*

```
*PIE CMD ERR
```

If you receive the message at left, return to step 1.

If you type a name longer than 30 characters, the CP will use only the first 30 characters as your filename. If you mistype your filename, type **CTRL-X** *to cancel the line and retype it correctly. The CP cannot edit text with the Text Editor commands.*

4. *Press* **RETURN**.

After the disk drive whirrs for a minute, the CP will save your file, report the following information, and await your next command:

```
FILENAME:

LENGTH:
MEMORY LEFT:


COMMAND?:
```

The number reported next to LENGTH: tells you how many characters your file has used. MEMORY LEFT: tells you how many more characters it would take for your file to fill up the text buffer.

CP Command Review	Function
S (or **SAVE**) **filename RETURN**	Saves, or writes, a file to diskette as "filename"

A Caution About File Lengths

If you are typing a particularly long file, the text buffer may approach its capacity. When this happens, you will be notified with a warning beep, and the Status Line will display the message "ALMOST OUT OF MEMORY." Each additional character you type will produce another beep. Stop typing at the next convenient point, such as at the end of the paragraph, and save your file on the diskette. You can then continue your document in a new file under another, closely related, filename, such as ANYFILE JR.

It is EXTREMELY important that you do not type more than another 21 lines. Doing so will cause the buffer to run completely out of memory space. Your typing will overwrite the disk operating system, and you will not be able to save any of your file.

Again, when you get the "ALMOST OUT OF MEMORY" warning:

1. *Stop typing as soon as conveniently possible.*
2. *Save the file.*
3. *Continue your document in a new file with a different, but appropriate, name.*

A word of reassurance: You are unlikely to type on forgetfully after the first warning, since the beep will sound with each additional character you type.

Editing a File

Now that you have saved your file, you have two copies of the text: one on the diskette, and one still in the text buffer. To edit your text, you must return to the on-screen PIE Text Editor.

Getting the file back on the screen from the text buffer is practically instantaneous. After saving your file, the file status is redisplayed as shown below at left. Respond to the prompt with the commands on the right.

```
FILENAME:        ANYFILE
LENGTH:                  x
MEMORY LEFT:     22015 - x

COMMAND?:
```

1. Type **E** (or **EDIT**)
2. Press **RETURN**

Your text is back on the screen.

CP Command Review	Function
E (or **EDIT**) **RETURN**	Returns you to the Text Editor to review current file for editing

Saving an Edited File

Here is an important concept to grasp: If you make changes in a file, you must save the file again. Otherwise, the changes will not be recorded and will disappear the first time you clear the contents of the text buffer.

Now make some changes or additions to your text that is already on the screen. If you have loaded the HELP file using the **%H** at the Command Processor level, you need only press **ESC H RETURN** to display all the simple commands and their functions if you need help remembering any editing or scrolling commands. Press **RETURN** to get back to the point in the file where you left off.

When you are ready to save your changes, try the next exercise.

Modifier	Command Key (Label)	Function
ESC H	**RETURN**	Displays all simple Text Editor commands and their functions; press return to continue

Displaying Your Filename

When you save an edited file that contains changes of any sort, you may want to save the changed file using the same name you gave the original file. Otherwise, you will be saving two files: one with changes and one without.

Whenever you return to the CP from the Text Editor, your filename is displayed. You can ask the CP to display the current filename by typing **? RETURN** in response to the COMMAND?: prompt.

Now, repeat the same steps you performed the first time you saved the file (see p. 23). The edited version of your text replaces the earlier version of the file on the diskette. The next time you load this file from the diskette, all your editing changes will be incorporated.

CP Command Review	Function
? RETURN	Displays current filename

Viewing the Catalog of Files

Prove to yourself that the file has been saved on the diskette. Clear the text buffer by turning the computer off, then on again. (If your diskette does not boot automatically, see "Getting Started" on p. 7.) After the diskette boots, the System Menu will appear. Press **1** to enter the Command Processor.

You can get the CP to provide you with a list of all the files stored on a diskette. Each diskette contains a "catalog," or index, of all its files. This catalog includes both the files that enable the PIE Writer system programs to run as well as your document files.

To get the CP to display the catalog on the screen, respond this way to the prompt:

1. *Type* **C** *(or* **CATALOG***)*
2. *Press* **RETURN**

If you have more than one disk drive, type **C,D1** or **C,D2** to catalog the desired diskette.

The list of files appears to the right of the file type abbreviations and file lengths. It should be easy to find the name of your file on the list.

CP Command Review	Function
C (or **CATALOG**) **RETURN**	Catalogs the current diskette; type **C,D1** or **C,D2** instead of **C** if you have more than 1 disk drive

Loading a File

Load your file (if you have not already), in response to the CP prompt, to see if the edited version was remembered by PIE Writer.

1. *Type* **L** *(or* **LOAD***), followed by one blank space and the name of your file (typed EXACTLY as listed in the catalog)*
2. *Press* **RETURN**

The name of your file reappears on the screen.

If you mistype your filename, you will be informed that the file is not on the diskette, with the computer's response:

```
FILE NOT FOUND
```

If this happens, simply retype the filename after the next COMMAND?: prompt. The CP is obedient, but literal. You must always be precise.

CP Command Review	Function
L (or **LOAD**) **filename**	Loads a file into the **RETURN** text buffer

PIE Summary

In section 2, you have learned:

1. *Basic cursor movement, window control, and most simple Text Editor commands.*
2. *How to begin a new file in the Command Processor.*
3. *How to save a file.*
4. *How to edit a file in the text buffer.*
5. *How to save an edited file.*
6. *How to display a filename.*
7. *How to display a catalog of the files on a diskette.*
8. *How to load a file from the diskette.*

All the basic PIE commands already introduced and all the advanced PIE commands are fully documented in section 5, "Reference Section," on p. 79. Also, all the PIE commands are noted in an abbreviated form on the PIE Writer Reference Card.

Now that you know how to create a perfect document with PIE, the next step is to learn how to print it. The FORMAT Text Processor controls the layout and printing of the documents you create with the PIE Text Editor and Command Processor.

3

The FORMAT Text Processor

The FORMAT Text Processor is the third part of the PIE Writer Word Processing System. It controls the layout, design, and printing of your document.

There are two parts to FORMAT. The first part lets you control the layout and design of your document by entering special commands right along with your text while you are working with the Text Editor. The second part of FORMAT lets you specify printing options, such as which document you want to print, the number of copies you want printed, and whether you want to print all or only certain pages.

In this section, you will learn how to use FORMAT dot commands to specify the appearance of a document and FORMAT Text Processor commands to print it out.

LESSON 1: INTRODUCTION

To specify the layout or design of your document, you enter FORMAT dot commands with the rest of your text while working on the PIE Text Editor screen. The instructions appear on the screen as you type them. Later, you instruct the FORMAT Text Processor to execute them.

Entering FORMAT Dot Commands

When entering FORMAT dot commands while working on the PIE screen, every command must:

- *begin with a dot*
- *begin at the left margin*
- *be sole occupant of an entire line*
- *precede the text it applies to*

Commands may be entered in either uppercase or lowercase.

For practice in entering FORMAT commands, load the sample file named COLUMNS, included on your PIE Writer diskette. (If you are currently practicing on the PIE screen, exit to the CP with a **CTRL-SHIFT-P**.) The CP will prompt:

COMMAND?:	*Load the practice file by typing* **L COLUMNS**
COMMAND?:	*To see the COLUMNS file on the PIE screen, type* **E**

When you have COLUMNS on the screen:

1. *Position the cursor on the "A" in "ADJECTIVES"*
2. *Press* **CTRL-I** *(a new Text Editor command about which you will learn more in section 5) to insert a blank line*
3. *Type* **.NF**
4. *Position the cursor on the "S" in "STRONG"*
5. *Press* **CTRL-I**
6. *Type* **.FI**

When you are done, your screen should look like this:

.nf *initiates the "no-fill" mode*	.NF		
	ADJECTIVES	NOUNS	VERBS
	WINSOME	DOGS	EAT
	GIANT	BONES	GROW
.fi *begins the "fill" mode*	.FI		
	STRONG	DOUBTS	CRY
	SOFT	OATHS	FLOAT
	SMALL	LOANS	AFFECT
	SUBTLE	PLANS	SHIFT

Displaying a File with the FORMAT Text Processor

To see exactly how the **.fi** and **.nf** commands affect your text, you should review the "formatted" file.

Leave the Text Editor with a **CTRL-SHIFT-P**. You will be prompted by the CP:

`COMMAND?:`	*To summon the FORMAT Text Processor, you type* **F**

When the drive stops whirring, the Text Processor's first prompt appears on the screen:

`OUTPUT TO PRINTER...`	*For this exercise, press* **RETURN**

The Text Processor is asking you for a command. It wants to know where to send, or "output," the formatted text you are about to identify. You can output your file to a printer, a diskette, or to the screen. For this exercise, you selected the screen by pressing **RETURN**.

Another prompt follows:

`PAGE LIMITS...`	*To output the whole file, press* **RETURN**

FORMAT can output either certain pages of your document or the entire file. Here, the Text Processor asks which pages of the file you want it to work on. By pressing **RETURN**, you selected the whole file.

`INPUT FROM DISK(D) OR PIE MEMORY BUFFER(M)?`	*To input a file in the PIE memory buffer, type* **M**

FORMAT wants to know where to find the "input" file. Text you have just entered or loaded is stored in the memory buffer. For this exercise, the file COLUMNS, to which you just added FORMAT dot commands, is still in the memory buffer.

FORMAT can also find and load a file that is on the diskette. If your file was one already stored on diskette, you would have typed **D** in response to the last prompt. FORMAT then would ask you for:

FILENAME?

To get a file from diskette, type the **filename** *and press* **RETURN**

Before FORMAT can get a file from the diskette, you have to enter the name of the file you want loaded. (Here, since you typed **M** instead of **D**, FORMAT goes straight to work on COLUMNS, which is stored in the memory buffer.)

Remember that the commands you entered in the COLUMNS file are not yet saved on the diskette, but they are still in the memory buffer.

After you have told FORMAT where to find your text and, if on the diskette, what its name is, the cursor will flash at the bottom left-hand corner of your screen, waiting for your next command. Tap the **space** bar once.

When you tap the **space** bar, formatted text from the COLUMNS file begins scrolling from the bottom of the screen to the top very quickly. You can tap the **space** bar to stop the scrolling, then tap it again when you are ready to advance the text.

The screen will stop scrolling automatically after displaying 24 lines of formatted text. It also stops at the end of every formatted page (you will learn how to specify the length of your pages with a dot command later), and displays a series of dashes, just above where the cursor flashed. Again, tap the **space** bar to advance the text.

When the entire file has been displayed on the screen, the flashing cursor again appears in the bottom left-hand corner.

Reading Formatted Text on the Screen

For users with a standard PIE Writer system, the formatted text scrolls up the screen as shown below when you tap the **space** bar:

```
ADJECTIVES    NOUNS    VERBS

WINSOME       DOGS     EAT
GIANT         BONES    GROW
STRONG DOUBTS CRY SOFT OATHS FLOAT SMALL
  LOANS AFFECT SUBTLE
PLANS SHIFT
```

Note the effects of the **.nf** and **.fi** commands on the text in the COLUMNS file.

The "fill" mode is a FORMAT default, so formatted text is always displayed in fill mode when it is sent to the screen, printer, or diskette, unless otherwise instructed with the **.nf** command. In the fill mode, FORMAT normally gathers enough characters (and spaces) to fill a 65-character line of output text, without breaking words in two. When a file is printed in fill mode, each line contains no more than 65 characters, regardless of the number of characters each line on the screen contains.

The **.nf** command you inserted at the beginning of the file instructs FORMAT to stop filling, initiating the "no-fill" mode. No-fill lets you enter tabular material and output it as it appears on the screen. This allows the first three lines of COLUMNS to remain within actual columns.

The **.fi** command reinstates the fill mode, forcing the last four lines of column material to run together as if they were part of a paragraph.

The formatted line beginning with "LOANS" is appended to the line ending with "SMALL" in the 80-column version of PIE Writer. The line breaks after "SMALL" in the standard version of PIE Writer because that version can display only 40 characters per line on the screen. When the text is printed, though, both the 40- and 80-column versions print the words from "STRONG" to "SUBTLE" on a single line.

Other FORMAT Commands

There are several other simple FORMAT commands covered in this section. These and more sophisticated FORMAT commands are covered in detail in section 5 and are listed in abbreviated form on the Reference Card.

If no command is entered to alter one of FORMAT's "permanent" initial values, then those values go into effect automatically. (Instructions on how to alter "permanent" FORMAT values are in section 4, "PIE and FORMAT Configure.")

Most FORMAT commands take an 'n' value, however, allowing you to alter the initial "default" values temporarily. For instance, the **.ll** command allows you to alter the 65-character default value for line length initially set up by FORMAT. Entering an 'n' value immediately following the **.ll** command lets you make your printed document 'n' characters wide.

Finally, some FORMAT commands force the text that follows them to "break" off and begin a new line of output text, even when they are entered in the fill mode.

It is important to keep these concepts in mind if you decide to alter the "permanent" FORMAT values, which are now set up for typical file formatting. Changing values for certain parameters can make formatting a specific type of file much simpler.

LESSON 2: PRINTING A FILE

Transferring Between the Text Processor, the Text Editor, and the System Menu

To practice commands or enter text on the Text Editor and print them out with the Text Processor, you should be able to transfer back and forth between the two easily.

To leave the Text Processor and return to the Text Editor, press the **space** bar in response to the flashing cursor that appears after the COLUMNS file is displayed on the screen until the FORMAT Text Processor prompts:

RETURN TO PIE (Y/N)?	*To return to the CP, type* **Y**

To summon FORMAT quickly from the CP mode, you must respond to the CP prompt:

COMMAND?:	*To summon FORMAT again, type* **F**

You can transfer to PIE when you are being prompted for input and output information by the Text Processor also. When any of the FORMAT prompts appear, respond with an **ESC**ape. For instance:

OUTPUT TO PRINTER...	*To summon the CP from FORMAT, press* **ESC**

Again, you are asked whether or not you want to:

RETURN TO PIE (Y/N)?	*To return to the CP, type* **Y**

If you would rather rerun FORMAT, type **N** or press **RETURN** in response to the above prompt. The prompt below will appear. Type **Y** or press **RETURN** to rerun FORMAT.

RE-RUN FORMAT (Y/N)?	*To rerun FORMAT, type* **Y** *or press* **RETURN**

Typing **N** in response to both the RETURN TO PIE and RE-RUN FORMAT prompts produces an endless loop of the same prompts over and over again until you either:

1. *Type* **Y** *in response to one of the prompts*
2. *Type* **%M** *in response to either prompt, which returns you to the System Menu*
3. *Type* **%Q** *in response to either prompt, which transfers you to BASIC*

Printing Hard Copy

You can send text to your printer, allowing you to compare "hard," or printed, copies.

A formatted copy is printed using FORMAT, with all the commands executed. A literal copy, showing text and commands exactly as you entered them, is printed using the Command Processor.

To print a formatted copy of COLUMNS, with FORMAT commands executed:

1. *Make sure the printer is on and "on-line"*
2. *Summon the CP*
3. *Load COLUMNS, if it is not in the text buffer already*
4. *Type* **F** *and press* **RETURN**
5. *Respond to the FORMAT prompts as follows:*

`OUTPUT TO PRINTER...`	*Type* **P**
`PAGE LIMITS...`	*Press* **RETURN**
`INPUT FROM DISK...`	*Type* **M**

The printer should start printing, producing formatted copy as shown below:

```
ADJECTIVES          NOUNS          VERBS


winsome             dogs           eat
giant               bones          grow
strong doubts cry soft oaths float small
 loans affect subtle
plans shift
```

If your printer does not begin printing, check all your hardware connections and run through the five steps above once more. If your printer still does not work, you may need to reconfigure your system (see section 4, "PIE and FORMAT Configure" for a complete explanation).

To print a literal copy of COLUMNS, showing text and FORMAT editing commands exactly as you entered them:

1. *Make sure the printer is on and on-line*
2. *Summon the CP*
3. *Load COLUMNS, if it is not in the text buffer already*
4. *Respond to the COMMAND?: prompt by typing* >**#1**
5. *Press* **RETURN**

If your printer card is not in slot #1, enter the card's correct slot number in place of the "1" in step 4.

The printer should produce a literal copy of COLUMNS, as shown below:

```
.NF
ADJECTIVES          NOUNS          VERBS


WINSOME             DOGS           EAT
GIANT               BONES          GROW
.FI
STRONG              DOUBTS         CRY
SOFT                OATHS          FLOAT
SMALL               LOANS          AFFECT
SUBTLE              PLANS          SHIFT
```

Finally, to take advantage of all the Text Processor's features, refer to the full list of options described below:

Prompt	*Option*
OUTPUT TO PRINTER...	A file can be output to the printer, diskette, or screen. To output from 1 to 99 copies of a file to the printer, type the number of copies desired and press **RETURN**. To output a single copy of a file to the printer, diskette, or screen, press **P**,**D**, or **RETURN**, respectively.
PAGE LIMITS...	To print a single page in a file (for example, page 3), type **3,3** and press **RETURN**. To print pages 4 through 12, type **4,12** and press **RETURN**. To print from page 6 to the end of the file, type **6** and press **RETURN**. To stop printing before the beginning of a page, which is desirable when feeding single sheets of paper into the printer, type **S** after you have set your page limits. For instance, to print pages 2 to 5, stopping before each page, type **2,5S**. To stop before printing each page in a file, type **S**.
INPUT FROM DISK...	To get a file from the diskette, type **D**. To get a file still in the text buffer memory, type **M**. To get a catalog of a diskette's files, type **C**. You will be prompted:
ENTER S#,D#...	Enter the drive number holding the diskette whose files you want to catalog (if you are using more than one disk drive). Also, enter the slot number holding the Apple Disk Interface card, if applicable. If neither of these apply to your system, press **RETURN** to catalog the current drive.
FILENAME:	Enter the name of the file you want to work on.

LESSON 3: PARAGRAPH CONTROL

Standard Paragraphs

The command for a standard paragraph is one of FORMAT's most useful features. The **.pp** command signals the beginning of a paragraph. It also activates four other built-in commands, outlined below:

```
.PP
BEGINS A PARAGRAPH
```

.sp—*Space down 1 line*. Here, a .sp command leaves 'n' blank lines between the last line of text and the first line of the new paragraph. With the .pp set, there is 1 blank line between paragraphs (n = 1).

.ti—*Temporary indent 5*. A .ti is used to indent the first line of each paragraph 'n' spaces. The .pp command sets the temporary indent at 5 spaces (n = 5).

.ne—*Need 2 lines to bottom*. A .ne measures the space remaining on the current page as it has been set up for output. If there is not enough room for at least 'n' more lines of text, the new paragraph will begin on the next page. The .pp command sets n = 2.

.fi—*Fill*. The .pp turns on the fill mode.

Every time you insert a **.pp** command at the beginning of a paragraph, FORMAT automatically executes the other four commands.

The no-fill paragraph command, **.np**, begins a no-fill paragraph, which allows you to print text exactly as it appears on the screen. It does not set a temporary indent, but it does activate three other built-in commands, two of them the same as those affected above:

1. **.sp** *spaces down 1 line (***.sp 1***)*
2. **.ne** *has a "need" value of 2 (***.ne 2***).*
3. **.nf** *stops filling of text.*

The **.np** allows you to alter the need value by entering your own value, paragraph by paragraph. For example, **.np 5** sets the need value at 5.

Other Paragraph Commands

Additional paragraph control commands override values set by **.pp** or **.np**, as described below:

.ps—*Modify paragraph spacing*. This command overrides .sp 1 to increase or eliminate space between paragraphs.

.pn—*Lines needed to begin paragraph*. A .pn overrides the default .ne 2 to alter the number of lines needed to begin a paragraph at the bottom of the page. This is helpful in assuring that there is room for 'n' lines of a paragraph beginning at the bottom of a page.

.pi—*Paragraph indent*. This overrides the default .ti 5 to alter the first-line indent of each paragraph.

These three paragraph commands are activated by **.pp** or **.np**. Since a no-fill paragraph has no indentations, **.np** activates only **.ps** and **.pn**. A **.pp** will activate all three commands. Once entered, **.ps**, **.pn**, and **.pi** work every time you enter either the **.pp** or **.np** command to begin a paragraph.

For example, to set up:

- *3 blank lines between paragraphs*
- *an 8-space indent on the first line of paragraphs*
- *need room for 4 lines at end of page*

You enter:

```
.PS 3                          3 blank lines
.PI 8                          8-space indent
.PN 4                          Need 4 lines
.PP                            The pp must be
THESE ARE THE COMMANDS           entered last
YOU SHOULD HAVE ENTERED.
```

If you enter another **.pp** to start a new paragraph, it will be formatted just like the first one. To check, print the paragraphs using the Text Processor.

Once you have entered **.ps**, **.pn**, and **.pi** values, they are executed every time you begin a new paragraph. To return to the original default settings, load your practice file, then enter:

```
.PS
.PN
.PI
PARAGRAPHS NOW RETURN
TO NORMAL.
```

If you enter a **.pp**, type another paragraph, then print it, the return to the default values should be apparent.

One-Time Paragraph Indent

One final note about the **.pp** command: you can specify the first-line indent of a single paragraph by entering a 'n' value after the **.pp**. Otherwise, it works just like a plain **.pp**, activating **.sp**, **.ne**, and **.fi** commands.

For instance:

- **.PP**—*indents a single paragraph 5 spaces*
- **.PP 0**—*no indent*
- **.PP 10**—*indents a single paragraph to begin in 11th column*

FORMAT Command Review	Function
.pp n	Begin paragraph with optional first-line indent of 'n'
.ps n	Modify paragraph spacing
.pn n	Lines needed to begin a paragraph at the bottom of a page
.pi n	Set paragraph indent
.np n	Begin no-fill paragraph with optional one-time need value of 'n' lines

The next lesson shows you how to use the paragraph commands, along with other commands, to format a letter.

LESSON 4: DESIGNING A LETTER

If you have been composing and printing your letters on a typewriter, you may be used to doing page layouts by eye. When producing letters or any business

report with FORMAT, you should get used to specifying page layout by calculation and command. Now you will learn how to design a letter with normal dimensions:

- *the paper is 8.5 inches wide*
- *the print size is 10 characters per inch*

If your paper and print sizes are different, substitute your own measurements in the following calculations.

Calculating Margin Widths

On a centered page with equal left and right margins, each margin will be:

$$\frac{\text{page width} - \text{line length}}{2}$$

You know that the page is 8.5 inches wide. With FORMAT, it is easier to think in terms of character spaces. To convert inches to character spaces, just multiply the width of the page in inches by the characters per inch. In this exercise:

$$\begin{aligned}\text{page width} &= 8.5 \text{ (inches) x } 10 \text{ (characters per inch)}\\ &= 85 \text{ character page width}\end{aligned}$$

Next, you will need to specify the line length. For now, use FORMAT's standard line length. The standard line is 65 character spaces long.

Now that you know the page width and line length, you can figure out the margins:

$$\frac{85 \text{ (page width)} - 65 \text{ (line length)}}{2}$$

Each margin will be 10 spaces, or 1 inch, wide.

Setting Up the Left Margin

The **.po** (page offset) command helps you set up the left margin of your page. The space you allot for your left margin with page offset is not equal to the width of the left margin, however, for most printers.

Since the **.po** default is 0, you might suppose that, by default, the printer should begin printing at the left edge of the page. But, in fact, print begins only as far left on the page as the printer can manage. The area in which it cannot print becomes part of the left margin.

To determine the overall distance in which the printer cannot operate, print out any paragraph, like the paragraph below, preceded by a **.po 0**

command, then measure from the left edge of the paper to the beginning of the print.

```
.PO 0
PAGE OFFSET IS SET AT 0.
THE PRINTER MARGIN IS THE
SPACE TO THE LEFT OF THESE
LINES.  IN THIS INSTANCE,
IT IS EQUAL TO 5 CHARACTER
SPACES, OR 1/2 INCH.
```

In the example above, the printer leaves a half-inch margin before it begins to print. Half an inch equals 5 character spaces (print size is 10 characters to the inch). The left margin, then, needs 10 spaces. The printer margin donates 5 of them. The **.po** command's 'n' value is used to make up the difference.

10 (left margin) - 5 (printer margin) = 5 (**.po '5'**)

For a similar printer, the page offset 'n' would be 5 (or 10 minus the printer's margin width).

FORMAT Command Review	Function
.po n	Sets page offset

Line Length

The **.ll** command controls the number of character spaces per line, including indentations and blank spaces between words.

For this exercise, use the PIE Writer system default of 65 characters.

Right Margin

The right margin is the space left over when you subtract the left margin and the line length from the total page width.

10 (left margin) + 65 (line length) = 75

85 (page width) - 75 = 10 (right margin)

So, to set the margins for a centered page with standard 65-space lines:

The **.po** *command is all you need to enter.*

```
.PO 5
TEXT FOLLOWING HAS
65-CHARACTER LINES AND
1-INCH LEFT AND RIGHT
MARGINS.
```

Work on setting up pages of different text widths and different margins until you fully understand how they are set up with FORMAT.

More layout practice follows. Remember the page offset value appropriate for your printer.

The Heading and Date

The lines between the top of the page and the heading, of course, vary with the length of the entire letter so that the letter is centered vertically on the page. For this exercise, assume that 10 lines of space will be needed before the heading. If the letter turns out to be shorter or longer than expected, the space above the heading can be altered after reviewing the formatted letter before it is printed.

Assuming that 10 lines of space are appropriate for this letter, you enter:

Set the page offset	`.PO 5`
Space down 10 lines	`.SP 10`
Cancel the fill mode	`.NF`

Indent the heading and date to start at the middle of the page (33 characters)	`.IN 33`
	`UNIVERSAL MACHINES, INC.`
	`1000 MAIN STREET`
	`PARIS, TEXAS  00000`

Allow 1 line space to date	`.SP` `JANUARY 1, 1999`
Cancel indent	`.IN`

What you have done is set the page offset for your printer, spaced down 10 blank lines, then indented the heading and date to the middle of the page (33 is half the line length), allowing 1 line space between them.

The **.in** command has the following functions:

.in—*Indent.* The .in command indents all the lines that follow it 'n' character spaces to the right of the page offset (.po) value. The .in command works in both fill and no-fill modes.

A **.nf** command is entered on the third line in the screen above so that FORMAT outputs each line exactly as it is entered. Otherwise, FORMAT would gather enough characters to fill up a 65-character line, then begin the next line at the 33-character indent.

The **.in** command at the end of the text above allows the next line entered to begin flush at the left margin because it returns the indent default to zero.

FORMAT Command Review	Function
.sp n	Insert 'n' blank lines
.fi	Set fill mode
.nf	Set no-fill mode

Inside Address and Salutation

Now space down 4 lines and type the inside address, skip a line, and type the salutation:

Space down 4 lines	.SP 4
Type inside address	UNIVERSAL PARTS, INC. 9999 FOREST AVENUE ROME, NEW YORK 99999
Allow 1 line space to salutation	.SP DEAR LADIES OR GENTLEMEN:

Since you are still in the no-fill mode (**.nf**) and the indent (**.in**) has been returned to its default value, the inside address will be set line by line, each line flush at the left margin.

If you had not cancelled the PIE Writer system's default fill mode, the heading, date, inside address, and salutation, could be broken line by line with a **.br** command.

.br —*Break in filling*. In fill mode, the **.br** command tells FORMAT to stop filling and make the next line a new one.

Note that the **.br** *command instructs FORMAT to begin a new line of text, whereas the* **.sp** *command begins a new line of text AND inserts blank lines.*	.SP 4 UNIVERSAL PARTS, INC. .BR 9999 FOREST AVENUE .BR ROME, NEW YORK 99999 .SP DEAR LADIES OR GENTLEMEN:

Line breaks are built into many FORMAT commands. For example, when you enter a **.sp**, the line following the command breaks to begin a new line, rather than fill in at the end of the line above the **.sp** command.

FORMAT Command Review	Function
.br	Break in filling

Body of Letter

Return to the fill mode before entering any text so that the body of your letter will be printed in 65-character lines. The **.pp** command returns you to the fill mode and skips one line before beginning the first paragraph:

Begin new paragraph	.PP ENTER A PARAGRAPH HERE.
Begin new paragraph	.PP ANOTHER PARAGRAPH.

Now start typing paragraphs. Use the **.pp** command to insert a blank space between paragraphs, to break for a new line, and to indent the first line of the paragraph 5 spaces.

To make text entry easier, you can enter the PPWrap mode (see p. 00 for a review). Press **ESC** then **RETURN** for uninterrupted text entry.

The right-hand margin will not be justified automatically, but you can initiate the right justification of text with an **.ad** command:

.ad—*Adjust right*. This command increases the space between words to ensure that each line of text following the command is the same length when your document is printed. If your printer is capable of incremental spacing, and if you have specified one of the daisy wheel pritners as the printer type in FORMAT Configure, the **.ad** command initiates incremental spacing as well as right justification (see p. 75 for a full explanation of this reconfiguration procedure).

.na—*No adjust right*. This command cancels right-margin justification and incremental spacing.

FORMAT Command Review	Function
.ad	Adjust right
.na	No adjust right

Closing and Signature

Skip a line, then enter the closing and signature on the same indent as you did the heading and date. To do this, and to make sure that both closing and signature appear on the same page, use the **.np** command:

Allow 1 line space	`.NP 6`
Set indent	`.IN 33`
	`YOURS TRULY,`
Allow 4 lines of space to signature	`.SP 4`
	`YOUR NAME`

Note that the **.np** command returns you to the no-fill mode, in which lines are printed out as typed, each one starting after the 33-character indent.

Saving and Reviewing the Letter

Now leave the Text Editor with a CTRL-SHIFT-P, save the file containing your letter, then transfer to the Text Processor:

To save your letter, type **S FILENAME**	`COMMAND?:`
To transfer to the FORMAT Text Processor, type **F** *then press* **RETURN**	`COMMAND?:`

To see your formatted letter, you will want to output FILENAME to the screen:

To output FILENAME to the screen, press **RETURN**	`OUTPUT TO PRINTER...`
To output the entire letter, press **RETURN**	`PAGE LIMITS...`
Since you have saved the letter under FILENAME you can input from the buffer by typing **M** *or you can input from the diskette by typing* **D.**	`INPUT FROM DISK...`

If you input from the diskette, type your **FILENAME**

FILENAME:

Tap the **space** bar to see your letter.

Notice that the indented parts of your letter, the heading, date, closing, and signature, are indented 33 character spaces on the formatted screen. Once the screen prints the 40th character on the line, the remaining text wraps around to the left side of the screen.

Now, print a copy of your letter to see how well you have done.

Other FORMAT commands can be used to alter the appearance of your printed letter:

.ls—*Line spacing*. Your letter will come out single-spaced unless you change the default of 1 with a .ls command. For instance, double-spaced output can be achieved with a .ls 2 command.

.bl—*Force 'n' blank lines*. If you wish to reserve a certain amount of space in your letter, up to one full page of text, you can reserve that space for text, pictures, or graphics that will be inserted later with a .bl command. The .bl works just like the .sp command, except that the .bl forces 'n' blank lines even if the letter must break to the new page first.

.pl—*Page length*. This command gives you the opportunity to change the page length from the default of 66 to a length of 'n.'

.bp—*Begin page*. The .bp command tells FORMAT to begin the text that follows the command on a new page.

.ne—*Lines needed to bottom*. The .ne command makes sure that if there is not enough room for at least 'n' lines of text at the bottom of the page, then the text that follows the .ne begins on the next page.

FORMAT Command Review	Function
.ls n	Set line spacing
.bl n	Force 'n' blank lines
.pl n	Set page length
.bp n	Begin page
.ne n	Lines needed to bottom

FORMAT

If you were to alter the text in your text file, perhaps by formatting the file to print double-spaced instead of single-spaced, you would have to return to the Text Editor, insert the desired dot commands, save the file, summon the Text Processor, and review the formatted file again to ensure that the vertical spacing was correct.

A formatted copy of your file, printed using the Text Processor (see p. 35 for instructions), and a literal copy, printed using the CP (see p. 36), are reproduced below. Check them to make sure you understand all the layout commands covered in this section.

```
                                Universal Machines, Inc.
                                1000 Main Street
                                Paris, Texas  00000

                                January 1, 1999


Universal Parts, Inc.
9999 Forest Avenue
Rome, New York  99999

Dear Ladies or Gentlemen:

Enter a paragraph here.

      Enter another paragraph.


                                Yours truly,


                                Your Name
```

```
.PO5
.SP10
.NF
.IN33
UNIVERSAL MACHINES, INC.
1000 MAIN STREET
```

```
PARIS, TEXAS  00000
.SP
JANUARY 1, 1999
.IN
.SP4
UNIVERSAL PARTS, INC.
9999 FOREST AVENUE
ROME, NEW YORK  99999
.SP
DEAR LADIES OR GENTLEMEN:
.SP
.FI
ENTER A PARAGRAPH HERE.
.PP
ENTER ANOTHER PARAGRAPH.
.SP
.IN33
YOURS TRULY,
.SP4
YOUR NAME
```

LESSON 5: INDENTATION

Extract Indent

To set off a quotation or other extracted material from the rest of a document, you may want to indent both the left and right margins an equal amount.

To indent each margin, for instance, 5 spaces, you must enter **.in 5** to indent the left margin and **.ll -5** ("minus" 5) to reduce the line length by 5 spaces, thereby indenting the right margin. Each of these indents would be cancelled and normal text margins would be resumed, of course, by entering an **.in** and **.ll** command, returning the indent to 0 and the line length to 65, their default values.

The following commands would have to be entered to print an extract:

Set left and right margins	`.IN 5`
	`.LL -5`
Allow 1 line space	`.SP`
Enter extracted text	`ENTER EXTRACT OR QUOTE.`
Resume normal left	`.IN`
and right margins	`.LL`
Allow 1 line space	`.SP`
Resume normal text	`ENTER NORMAL TEXT AGAIN.`

FORMAT Command Review	Function
.in n	Set indent
.ll n	Set line length

List and Outline Indent

The 'n' variable that accompanies most FORMAT commands allows you to enter any value, including a "plus" or "minus" value. For instance, when using the indent command (**.in**) immediately preceding a line you want indented, such as a list or outline entry:

- **.IN 5** *indents the text that follows it 5 spaces to the right of the page offset (***.po***)*
- **.IN +5** *indents 5 spaces to the right of the current indent*
- **.IN -5** *moves the indent 5 spaces to the left of the current indent*

.IN+5 → *Using "plus" indentations,*
.IN+5 → *you can indent*
.IN+5 → *outlines*
.IN+5 → *all the way*
.IN+5 → *across the page.*
.IN-10 ← *Using "minus" indentations,*
.IN-5 ← *you can shift*
.IN-5 ← *indentations*
.IN ← *back to the left.*

An indent without a plus or minus sign is set to the column number given. A "plus" indent is always added to the current indent, set originally by an indent (**.in**), or temporary indent (**.ti**) command. A "minus" indent is subtracted from the current indent.

For instance, if you wanted to indent an outline entry 5 spaces, in addition to the 10 space indent already set, then run a list of items, you could use the **.ti** command as shown below:

Set 10-space indent	`.IN 10`
Indent first line an additional 5 spaces	`.TI +5`
	`THE .TI COMMAND ADDS A TEMPORARY INDENT, INDENTING THE FIRST LINE OF THIS PARAGRAPH 5 ADDITIONAL SPACES. THE REMAINDER OF THE LINE IS FLUSH LEFT ON A 10-SPACE INDENT.`

Set indent to "hang" number of list item 1.

```
.TI -3
1. THIS COMMAND ALLOWS
YOU TO "HANG" THE NUMBER
OF A LIST ITEM AND ALIGN
THE ITEM ITSELF AT LEFT
WITH TEXT ABOVE AND
BELOW.
```

The **.in**, **.ti**, **.pi**, and **.pp** commands can be accompanied by plus and minus values.

As discussed previously, the **.pp** command has a built-in temporary indent equal to a **.ti +5** command:

Indent first line an additional 5 spaces

```
.PP
WHEN YOU INDENT A
PARAGRAPH WITH THE .PP
COMMAND, THE RESULT WILL
BE THE SAME AS IF YOU HAD
USED THE .TI+5 COMMAND.
```

You can enter a **.pi** command to change the first- line indent every time the **.pp** is used:

Set 10-space indent
Set temporary indent
Begin paragraph

```
.IN 10
.PI 10
.PP
BECAUSE .PI AND .IN HAVE
EQUAL VALUES, THIS
PARAGRAPH WILL NOT SHOW
AN ADDED FIRST-LINE
INDENT.  THE ENTIRE
PARAGRAPH WILL BE ON A
10-SPACE INDENT.
```

To indent the first line of the paragraph 10 additional spaces, you would have to change the **.pi 10** to **.pi +10**.

You can produce items whose numbers "hang" with the **.in**, **.pi**, and **.pp** commands also:

Set 13-space indent

```
.IN 13
```

Hang number or "outdent" first line 3 spaces

```
.PI -3
.PP
1. THESE COMMANDS WILL
MAKE THE ITEM NUMBER HANG
AND SET THE ITEM,
BEGINNING WITH THE WORD
"THESE," FLUSH LEFT ON A
13-SPACE INDENT.
```

The actual indentation from the left margin to the number 1 is 10 spaces (or, 13 minus 3).

To indent the same distance from the right margin, reduce the line length:

Reduce line length
Set 13 space indent
"Outdent" first line 3 spaces

```
.LL -10
.IN 13
.PI -3
.PP
1. THIS PARAGRAPH WOULD
HAVE A HANGING ITEM
NUMBER, AN ITEM FLUSH
LEFT ON A 13-SPACE
INDENT, AND RAGGED RIGHT
ON A 10-SPACE INDENT FROM
THE RIGHT.
```

FORMAT Command Review	Function
.ti n	Set temporary indent for next line

The final FORMAT unit covers format commands for producing running titles, page numbers, and centered, underlined, capitalized, or boldface text.

LESSON 6: RUNNING TITLES, PAGE NUMBERS, MARGINS, AND FINISHING TOUCHES

Running Titles

You can create running titles on every page of your document. Running heads often contain titles that reflect a page's content, along with a page number.

Other pertinent information, such as a date, can be included as well, as shown below.

1999 RUNNING HEAD 100

You can create a running head with the **.he** command:

.he—*Head title*. This command prints the title, which is entered on the same line after the command, one line from the top of every page of your document. If you change the title used in the running head, enter the .he command, then the new title, and that running head will appear on the second line of the next page.

To produce a running title at the foot of your page, use the **.fo** command:

.fo—*Foot title*. This command prints whatever text that is entered on the same line after the command, three lines from the bottom of every page of your document.

The **.tl** command allows you to insert a title anywhere on the text page:

.tl—*Flush left, center, flush right*. The title that follows the .tl command is printed once, exactly where it is entered in the text.

All three of these commands, the **.he**, **.fo**, and **.tl**, follow a certain pattern for title entry. Up to three "parts" of the running head can be specified, with each part separated by an apostrophe. Their position within the running head is shown below:

Part 1 Part 2 Part 3

Part 1 begins at the left margin. Part 2 is centered. Part 3 is aligned flush right.

To enter a three-part head title, you would enter, all on the same line:

1. *The* **.he** *dot command*
2. *The first apostrophe*
3. *Part 1 of your running head*
4. *The second apostrophe*
5. *Part 2 of your running head*
6. *The third apostrophe*

7. *Part 3 of your running head*
8. *The fourth apostrophe (optional)*

On the screen, it would look this way:

```
.HE'PART 1'PART 2'PART 3'
```
To produce a three-part head title

To enter the same title at the foot of the page, enter:

```
.FO'PART 1'PART 2'PART 3'
```
To produce a three-part foot title

To use only the first part of a title, just enter the first part:

```
.HE'PART 1'
```
To produce the first part of a head title

To use only Parts 2 and 3, enter:

```
.HE''PART 2'PART 3'
```
To produce only parts 2 and 3 of a head title

The second consecutive apostrophe indicates that Part 1 is to be left blank. Part 2 will be output in its usual place at the center of the page, and Part 3 will be flush right.

To use only Part 3 (for example, to make sure that a letter's date is right-justified), enter:

```
.TL'''JANUARY 1, 1999
```
To produce only the third part of a head title

FORMAT Command Review	Function
.he t	Set head title 't'
.fo t	Set foot title 't'
.tl t	Set 't' flush left, center, flush right

Page Numbers

You can have FORMAT insert page numbers for you, on every page of your document, by using the '%' sign. To have the current page number produced at the foot of every page in your file, follow the format below:

.FO ' ' %	To center your document page number at the foot of the page

FORMAT automatically replaces % with the current page number. The command is continuous, so this one entry is all you need to assign page numbers to the whole file automatically, from page 1 to the end of your document. Of course, you can have your page number appear flush left, centered, or flush right at the head or foot of every page.

Margins

With margin commands, you can move head and foot titles up or down on the page. There are four top and bottom margins that you can manipulate to regulate this movement.

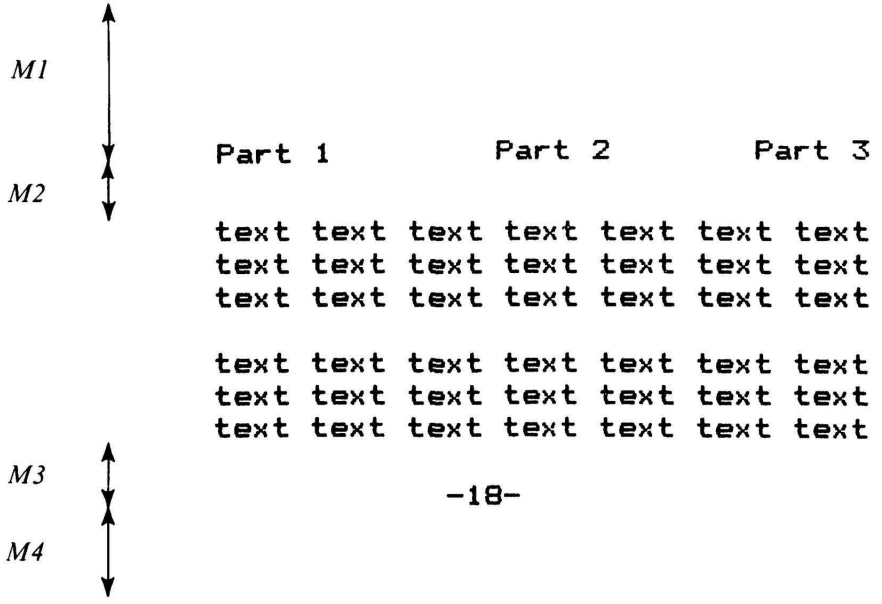
.m1—*First top margin (to head title)*. This command specifies the number of lines from the top of the page down to, and including, the head title. The default value for .m1 is n = 2.

.m2—*Second top margin*. This command specifies the number of lines between the head title and the first line of text, excluding the first line of text. The .m2 default is n = 2.

.m3—*First bottom margin (to foot title)*. The third margin specifies the number of lines between the last line of text and the foot title, excluding the foot title. The .m3 default is n = 1.

.m4—*Second bottom margin*. The fourth margin sets the number of lines from the bottom of the page, up to, and including, the foot title. The second bottom margin default is n = 3.

These four margins represent the space indicated below on a printed page:



To produce a three-part running head at the top of the page and a centered page number at the foot of the page, you would enter:

```
.HE'PART 1'PART 2'PART 3      Set the running head
.FO''-%-                      Set automatic page
.M1 4                           numbering
.M2 4                         Set margins
.M3
.M4 4
ENTER A FULL TEXT PAGE,       Enter text
WITHOUT RUNNING HEAD AND
PAGE NUMBER HERE.
```

The entry above produces the first page of a report with margins as illustrated below:

```
text text text text text
text text text text text
text text text text text
```

-60-

1 blank line here, the default value
3 lines below the page number to the end of the sheet

Of course, you can also set margins without using head or foot titles. Just turn off Margin 1 and Margin 4 by setting them at zero, which inhibits the output of head or foot titles. Set Margin 2 and Margin 3 wherever you want them.

```
.M1 0                    Ignore this margin
.M2 6                    Set margins 2 and 3 at 6
.M3 6                      each
.M4 0                    Ignore this margin
TEXT TEXT TEXT TEXT      Enter text
TEXT TEXT TEXT TEXT
TEXT TEXT TEXT TEXT
```

The commands entered above will produce 6 lines of space above and below the document text.

If you have no head and foot titles, you can produce a report with 4 blank lines at the top and bottom of each page by entering no margin specifications at all.

FORMAT Command Review	Function
.m1 n	Set first top margin (to head)
.m2 n	Set second top margin
.m3 n	Set first bottom margin (to foot)
.m4 n	Set second bottom margin

Finishing Touches

You can center and capitalize text anywhere on the page with the **.ce** and **.cp** commands:

.ce—*Center*. This command tells FORMAT to center the following 'n' lines of unformatted text. Whether it is used in the fill (.fi) or the the no-fill mode (.nf), the .ce command centers text broken line by line.

.cp—*Capitalize*. To capitalize for emphasis within text, use the .cp command, followed by the 'n' lines of text you want capitalized. This command is useful for entering subheads or for capitalizing a single word in the middle of a sentence, as shown below:

.FI	*Set the fill mode*
THIS IS THE WAY TO	*Enter text*
.CP	*Capitalize the word on the next line*
CAPITALIZE	
TEXT.	*Return to normal text*

If your printer supports it, you also can use the boldface and underline commands, **.bf** and **.ul**:

.bf—*Boldface*. The printer backspaces and restrikes text on the 'n' lines or changes to boldface font following the .bf command to achieve boldface text. As with the .cp command, the .bf can used to affect a single word in a sentence, making it bold.

.ul—*Underline*. To underline text, precede the 'n' lines of text you want underlined with the .ul command. The .ul command is capable of underlining even a single word in a sentence.

You may need to reconfigure FORMAT to use these commands with your printer (see p. 70–71 for details).

Note that the .cp, .bf, and .ul commands affect the entire line (or lines) that follow the command. To have these commands work on only a few words within a line, those words must be set off as a separate line when the text is entered. In fill mode, however, when those lines are formatted, they will take their proper place within the running text.

The following example demonstrates how all four of these commands can work together:

```
.NF                                  Set no-fill mode
.CE 2                                Center the next 2
CENTER THIS LINE.                      unformatted lines
.CP 3                                Capitalize the next 3
CENTER AND CAP THIS LINE.              unformatted lines
.BF                                  Boldface the next line
BOLDFACE AND CAP THIS LINE.
.UL                                  Underline the next
UNDERLINE AND CAP THIS LINE.           line
```

Note how a sense of economy in entering the dot commands above saved unnecessary typing.

FORMAT Command Review	Function
.ce n	Center the next 'n' lines
.bf n	Print the next 'n' lines in boldface
.ul n	Underline the next 'n' lines
.cp n	Capitalize the next 'n' lines

Read and work through the explanations of the commands covered until you have them mastered, then use the Reference Card to refresh your memory from time to time. All of the FORMAT commands, including those covered in this section and the more sophisticated commands not yet covered, are documented in section 5, "Reference Section," on p. 79.

Section 4, "PIE and FORMAT Configure," discusses how to set up the PIE Writer system permanently to accommodate your normal printer and page appearance needs. You do not need to change any of the default values in PIE Configure or FORMAT Configure, however, to begin creating and printing error-free documents.

4

PIE and FORMAT Configure

The system provided with each PIE Writer package, the diskette and documentation, is a complete word processing system that operates smoothly with almost any printer or Apple peripherals you may own.

The only information missing from the PIE Writer system is your computer's hardware configuration and printing needs. More precisely, PIE Writer needs to know specifics about what printer you are using, how you want to display characters on your screen, what page appearances meet your typical needs, and so on. The configuration files supplied on your PIE Writer diskette, "PIE CONFIGURE" for the editor (choice **3** on the System Menu) and "FORMAT CONFIGURE" for the text formatter (choice **4**), allow you to customize the system to match the hardware you are using and the appearances you prefer, both in editing and printing.

Do not reconfigure the original PIE Writer system diskette. Use the backup of the system diskette that you made after booting the original diskette the first time to reconfigure PIE and FORMAT.

Throughout this section, look at the options displayed on the screen as you read along in the documentation.

PIE CONFIGURE

When you wish to inspect or alter your present PIE configuration, select option **3** from the main System Menu. This copies the PIE program code to computer memory and allows you to review PIE Configure's default values, the values that PIE Writer uses unless you change them.

Configure

After you have chosen option **3** from the System Menu, the following screen appears:

```
1 - PRINTER INTERFACE
2 - LOWER CASE AND SPECIAL CHARACTERS
3 - OTHER PIE EDITOR DEFAULTS
4 - PROGRAM DISK SLOT, DRIVE, VOLUME
5 - DOUBLETIME SPOOLER OPTION


DO YOU WANT TO LOOK AT OR MODIFY VALUES
IN ONE OF THE ABOVE CATEGORIES?
```

If you respond **N** (for no) to the prompt, you are transferred to the main System Menu. If you respond **Y** (for yes), you receive another prompt:

```
WHICH ONE?
```

Instead of responding with **Y** or **N**, you can simply press the number corresponding to the line whose series of default values you wish to alter. Then, when prompted, enter the new value or select a new setting from the choices presented.

Printer Interface

Select **1** to view the printer interface options:

```
1 - PRINTER SLOT...............1
2 - PRINTER INITIALIZATION....NONE
3 - HIGH BIT SET ON OUTPUT....YES
4 - LINE FEED AFTER RETURN....NO

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

Before responding to the prompt above, read the descriptions of each option below:

```
1 - PRINTER SLOT...............1
```

As shipped, the PIE Writer diskette assumes that your printer interface card has been installed correctly in slot 1. If for some reason your interface card is not installed in that slot, you can reconfigure PIE to alert the system that the card is in another slot. Enter any number from **2 through 7** to change the default.

```
2 - PRINTER INITIALIZATION....NONE
```

If your printer controller card or printer requires a character or character sequence initialization, you may enter up to seven characters here. Each time PIE is executed, these characters are sent to the printer just before the actual text.

```
3 - HIGH BIT SET ON OUTPUT....YES
```

Some printers also require that the high-order bit in every 8-bit byte (bits and bytes are elements in the computer's machine language) be set on output before printing can commence. PIE's default setting covers this circumstance. Type **N** (for no) to change the default.

```
4 - LINE FEED AFTER RETURN....NO
```

Finally, some printers cannot produce line feeds after carriage returns unless they are instructed to do so by the software. PIE assumes that your printer can handle line feeds after carriage returns on its own. If your printer cannot, type **Y** (for yes) to instruct PIE to send this line feed command to the printer.

When you are finished making printer interface value changes, type **N** (for no) in response to the prompt to return to the PIE Configure menu.

Lowercase and Special Characters

Select **2** to view the lowercase and special character options:

```
1 - OUTPUT LOWER CASE TO CRT..NO
2 - INITIAL LOWER CASE INPUT..YES
3 - SHIFT KEY MOD.............NONE
4 - HIGH BIT SET IN BUFFER....YES
5 - CONVERT BREAK LINE TOKEN..[Z]

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

1 - OUTPUT LOWER CASE TO CRT..NO

The standard Apple II video display cannot show lowercase letters properly. Standard uppercase letters are displayed in white on a black background to represent lowercase. "Inverse" characters, black letters on a white field, represent uppercase letters. If your system includes a lowercase adapter, you must change this PIE default. Type **Y** (for yes) to display true uppercase and lowercase characters.

2 - INITIAL LOWER CASE INPUT..YES

Keystrokes are normally entered as lowercase letters, with uppercase accessed by using the ➡ (CASE) key. Type **N** (for no) to enter letters initially in uppercase and access lowercase letters with the same ➡ key.

3 - SHIFT KEY MOD.....NONE

There is a hardware modification that can be made to the Apple II that allows the **SHIFT** keys to function "normally," as on a regular typewriter. Even if the hardware modification is made, it will not be recognized by PIE Writer unless the proper setting is made here.There is a detailed discussion of this hardware modification on p. 149. Type **1** for GAME I/O, **2** for ROMPLUS (and the slot of the ROMPLUS card), and **3** for no modification.

4 - HIGH BIT SET IN BUFFER....YES

This indicates whether the high-order bit of each character in the text buffer is set to 1 or 0. For all but special applications, use the default (YES). Change the default by typing **N** (for no).

5 - CONVERT BREAK LINE TOKEN..Z

When loading a file that is known to have or may have lines longer than the text editor line length of 64 {128}, you can use a Command Processor auxiliary command (**%CB**) to shorten the lines in a file to a specific column value. When the lines are shortened, a CTRL-Z is inserted at the end of each broken line. Enter any control character here to change the break line token.

When you have finished making changes to the lowercase and special character options, type **N** (for no) in response to the prompt to return to the System Menu.

Other PIE Editor Defaults

Select **3** to view the other PIE Editor default options:

```
1 - PIE EDIT BUFFER END.......$95FF
2 - INITIAL BELL COLUMN.......38
3 - INITIAL TAB INCREMENT.....8
4 - PPWRAP INITIALIZATION.....OFF

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?

1 - PIE EDIT BUFFER END.......$95FF
```

In order for you to enter text characters into the computer, an area of the computer's storage memory, the text buffer, must be reserved for character storage. On very long documents, you can "fill up" the buffer area by continuing to key in characters until all of the reserved space has been filled. When the text buffer is filled, the Status Line displays an "ALMOST OUT OF MEMORY" message, indicating that it is time to save a copy of your file on diskette.

The text buffer begins at $4000 {$4C00 for 80-column board systems} and normally ends at the highest available memory location below the Disk Operating System (DOS), which is $95FF. You may want to decrease this value by entering another hexadecimal address if you need to reserve space for a large printer driver or other utility program.

The majority of users will never need to alter this setting.

```
2 - INITIAL BELL COLUMN.......38
```

If you were using the Text Editor right now, you would see the bell column marker, a reverse dash, in column 38 {78 for 80-column systems}, where it is indicated below:

```
-+-------+-------+-------+-------+---⊟-
!                                   ^!
!                                   ^!
!                                    !
----------------------------------------
```

The bell column marker indicates the position in the line at which either of the following occurs:

1. *In Manual Mode, the bell tone beeps to indicate that the cursor has reached the end of the first window. The bell tone beeps again at the end of the right-hand side of the window simply to remind the operator to press* **RETURN**.

2. *In PPWrap Mode, the bell column will not produce the bell tone. Instead, it will move the cursor down to the next line, positioned beneath the first printable character of the previous line. If this occurs in the middle of the word, all characters back to the last space character will be moved to the next line as well.*

Enter any value from **1** to **64** {or **128** for 80-column users}, then press **RETURN** to set a new bell column marker.

```
3 - INITIAL TAB INCREMENT.....8
```

Initially, the tab markers are set 8 spaces apart, beginning with the first space. Enter any value from **1** to **64** {or **128** in 80-column versions} to change this setting from the default to one you prefer, then press **RETURN**.

```
4 - PPWRAP INITIALIZATION.....OFF
```

As shipped, PIE Writer's Wrap mode is turned off. To change the default (0), type **1** for on, **2** for indent mode (see p. 89 for a full explanation of the auto indent feature).

When you have finished changing as many PIE Editor defaults as you wish, type N (for no) to return to the PIE Configure main menu.

Program Disk Slot, Drive, Volume

Select **4** to view the diskette slot, drive, and volume options:

```
1 - SLOT.......................6
2 - DRIVE......................1
3 - VOLUME.....................0

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

The slot, drive, and volume options are for users with more than one drive. These options tell the system where to find the system diskette. The system diskette must always be in the slot, drive, and volume specified here whenever you transfer between PIE and FORMAT or when the auxiliary files are accessed.

```
1 - SLOT.......................6
```

If the disk interface card attached to the drive you are using for your system diskette does not occupy slot 6, type the number of the slot in which the card is installed, from **1** to **7**.

```
2 - DRIVE......................1
```

If you wish to use DRIVE 2 for your system diskette, change this option by typing **2**.

```
3 - VOLUME.....................0
```

The volume should be set to 0 unless you have a hard disk system that requires a volume specification.

Type **N** (for no) in response to the prompt when you have finished changing system defaults for diskette slot, drive, and volume.

Doubletime Spooler Option

Select **5** if you have purchased the Doubletime Spooler:

```
DOUBLETIME INSTALLED..........NO

DO YOU WANT TO CHANGE THE ABOVE VALUE?
```

Type **Y** (for yes) in response to the prompt above if you are using the Doubletime Spooler. You must have transferred the "Doubletime Printer" file from that package onto your system diskette to use the spooler.

Those are all the PIE Configure options. Change the appropriate settings to suit your typical needs, then answer **N** (for no) to the prompt:

```
DO YOU WANT TO LOOK AT OR MODIFY
ADDITIONAL VALUES IN ONE OF
THE ABOVE CATEGORIES?
```

Another prompt appears:

```
DO YOU WANT TO PERMANENTLY
UPDATE PIE?
```

Answer **Y** (for yes) to store an updated copy of PIE on the system diskette. Your changes are "permanent" in that the system will begin each session with whatever values you select.The next time you boot the system, it will remember your preferred values. You can change these permanent settings, however, by selecting the PIE Configure option on the System Menu as often as you wish.

FORMAT CONFIGURE

This section of the PIE Writer program allows you to set up FORMAT to take advantage of your printer's specific features.

Make sure that you are about to reconfigure your backup—not your original—system diskette, then select **4** in response to the System Menu prompt. The following screen appears:

```
1 - PRINTER INTERFACE
2 - LOWER CASE AND SPECIAL CHARACTERS
3 - PAGE FORMAT (SPACING, LINE LENGTH)
4 - TOP AND BOTTOM MARGINS
5 - PARAGRAPHS AND FILLING
6 - PROGRAM DISK SLOT, DRIVE, VOLUME

DO YOU WANT TO LOOK AT OR MODIFY VALUES
IN ONE OF THE ABOVE CATEGORIES?
```

Again, walk through the options on your screen while reading along in the documentation.

Printer Interface

Select **1** to view the printer interface options:

```
1 - PRINTER SLOT...............1
2 - PRINTER TYPE...............OTHER
3 - PRINTER INITIALIZATION....NONE
4 - HIGH BIT SET ON OUTPUT....YES
5 - LINE FEED AFTER RETURN....NO
6 - UNDERLINE CAPABILITY......BS
7 - BOLDFACE CAPABILITY.......BS
8 - PAGE STOP ENABLED.........NO

DO YOU WANT TO CHANGE ONE
ONE THE ABOVE VALUES?
```

```
1 - PRINTER SLOT..............1
```

If you have a standard interface card installed in slot 1, FORMAT Configure is set up correctly for your system. If you do not have a standard interface card or if your card is not in slot 1, then type **1**, and the prompt below appears:

```
DO YOU HAVE A STANDARD
APPLE II INTERFACE CARD?
```

Respond by typing **Y** (for yes) if you have a standard card. The next prompt asks you to enter the card's slot number:

```
IN WHICH SLOT #
```

Type the slot number, from **1 through 7**, and FORMAT Configure will adjust the printer driver address for you.

If you do not have a standard card, respond **N** (for no) to the first prompt. Then enter the printer driver address in response to the prompt below, and press **RETURN**:

```
ADDRESS OF YOUR PRINTER DRIVER?

2 - PRINTER TYPE..............OTHER
```

As shipped, PIE Writer is ready to function as a word processing system with most printers commonly available on the market. FORMAT can be reconfigured to take advantage of your printer's maximum capabilities, however. Type **2** to receive the display below:

```
1 - DIABLO 1620          4 - EPSON MX-80
2 - QUME SPRINT 5        5 - CENTRONICS 737
3 - NEC SPIN 5510        6 - OTHER

SELECT PRINTER TYPE:
```

If your printer is one of the five listed, type the number to the left of the printer's name in response to the prompt. FORMAT Configure alters the remaining five values (items 3 through 7) in the printer interface option, allowing PIE Writer to take maximum advantage of your printer's capabilities. Users experienced with these printers may override any of the six values set up after the printer type is selected.

Selecting one of the first three printers displays the prompt:

```
# OF CHARACTERS PER INCH:
```

Enter the appropriate number and press **RETURN** to reconfigure FORMAT for any of those three printers.

The fourth and fifth printer options automatically reconfigure FORMAT without any further prompting.

When the sixth printer option is chosen, FORMAT Configure provides the prompt below:

```
CAN YOUR PRINTER BACKSPACE?
```

Respond by typing **Y** (for yes) or **N** (for no). FORMAT sets the sixth and seventh printer interface parameters based on your answer.

Again, the remaining five printer interface defaults can be overridden to accommodate your printing needs.

```
3 - PRINTER INITIALIZATION....NONE
```

If your printer controller card or printer requires a character or character sequence initialization, you may enter up to seven characters here. Each time FORMAT is executed, these characters are sent to the printer just before the actual text.

```
4 - HIGH BIT SET ON OUTPUT....YES
```

Some printers require that the high bit be set for proper operation.

```
5 - LINE FEED AFTER RETURN....NO
```

This setting varies from printer to printer. If FORMAT prints out the document all on the same line, then the answer here should be **Y** (for yes). Many printer controller cards automatically provide a line feed when the carriage return character is passed.

```
6 - UNDERLINE CAPABILITY.......BS
```

Type **0** if your printer cannot underline text using any of the methods described below.

BS indicates that FORMAT will implement the **.ul** command with a combined "backspace" and then "_" to create an underline. Type **1** to select backspace.

S/S indicates that your printer has an automatic underline capability (such as on the Centronics 739). Start/stop underlining is initiated with the control character sequence you define after typing **2** to receive the next prompt:

```
(ENTER CTRL CHARS USING CTRL-SHIFT-M)

UNDERLINE START SEQUENCE:
UNDERLINE STOP SEQUENCE:
```

Press **RETURN** after typing the underline sequence.

OVRLAY is used with printers such as the Epson, which cannot backspace or use start/stop, and is selected when you type **3** for this option.

```
7 - BOLDFACE CAPABILITY.......BS
```

Type **0** if your printer cannot produce boldface copy using the two methods described. Type **1** if you selected BS in the underline capability option. Type **2** if your printer uses a start/stop sequence to produce boldface print. You are prompted for the start and stop sequences.

```
8 - PAGE STOP ENABLED.........NO
```

Some printers require or allow single-sheet paper. If you are using single-sheet paper, you want the output to halt after each page is printed so that you can feed a new sheet in. Select **Y** (for yes) to accomplish this.

When you are finished reconfiguring the printer interface, return to the main FORMAT Configure menu by answer **N** (for no) in response to the DO YOU WANT... prompt.

Lowercase and Special Characters

Select **2** on the main FORMAT Configure menu to reconfigure the way lowercase and special characters are entered on the keyboard and appear on the screen.

```
1 - OUTPUT LOWER CASE TO CRT..NO
2 - INITIAL LOWER CASE (.GL)..YES
3 - SHIFT KEY MOD.............NONE
4 - ESCAPE CHARACTER..........']'
5 - BLOACK TERMINATING CHAR...'<'

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

You can reconfigure each of the following options by selecting the appropriate number.

```
1 - OUTPUT LOWER CASE TO CRT..NO
```

If you have a lowercase adapter, enter **Y** for yes in response to the prompt.

```
2 - INITIAL LOWER CASE (.GL)..YES

3 - SHIFT KEY MOD............NONE
```

These two options refer to text entry via the **.gl** command. They should be set to the same values as those chosen in PIE Configure.

```
4 - ESCAPE CHARACTER..........']'
```

Using an escape character while in PIE allows you to enter special format commands within a file. You can change the escape character from the default (]) to any character you prefer.

```
5 - BLOCK TERMINATING CHAR....'<'
```

The block terminating character is entered at the end of a data block in a secondary file when merging files to produce form letters, mailing labels, and so on. The '<' character is the default. Enter any character you prefer to change the default.

```
6 - LF AFTER CR OUTPUT TO DISK.NO
```

As stated previously, some printers rely on software to produce line feeds with each carriage return after a printed line. If this is the case with your printer, and you intend to print a file after first writing it to diskette, change the default by typing **Y** (for yes). Files printed from a diskette will have the line feed embedded automatically after each carriage return.

```
7 - STOP CHAR OUTPUT TO DISK..[C]
```

The printer stop escape,]+, must be cancelled if outputting a file to diskette. The]+ is translated to CTRL-C, which is recognized by the Double-time Spooler.

To return to the main FORMAT Configure menu after reconfiguring the lowercase and special character settings, type **N** (for no).

Page Format

To change the default format of your page, including spacing and line length, select **3** from the main FORMAT Configure menu to reveal the screen below:

```
1 - LINE SPACING...............1
2 - PAGE LENGTH................66
3 - LINE LENGTH................65
4 - PAGE OFFSET................0
5 - INITIAL OFFSET CHAR.......' '

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?

1 - LINE SPACING...............1
```

The default is set for single-spaced printouts. Select **2** for double-spaced or **3** for triple-spaced reports. You can enter any number up to one less than the line length, then press **RETURN**.

```
2 - PAGE LENGTH................66
```

FORMAT assumes an 8 1/2 x 11-inch report page, and 6 printed lines per vertical inch. Enter any value you want here from **1 to 250**, then press **RETURN**.

Note that this value is equal to the number of lines that can be printed on one of your pages, not necessarily the number of lines you want printed.

```
3 - LINE LENGTH................65
```

FORMAT assumes a 65-character line based on a 6 1/2-inch line and a 10 character-per-inch printer. The same line length on a 12 cpi printer requires a 78- character line. Enter whatever value suits your needs, from **1** to **132**, then press **RETURN**.

```
4 - PAGE OFFSET................0
```

Set the page offset value appropriate for your printer permanently here (see p. 00 to determine that value). Any value from **1** to a maximum of one less than the line length value is valid. Enter the page offset and press **RETURN**.

```
5 - INITIAL OFFSET CHAR.......' '
```

If you want to precede every line in your file with a character that your printer recognizes, enter the character here, and press **RETURN**. The default value is a blank space, for normal use.

Type N (for no) in response to the prompt when you have altered all the page format defaults you want.

Top and Bottom Margins

FORMAT Configure lets you change the head and foot margins permanently. Select **4** to see the following screen:

```
1 - MARGIN 1 VALUE............2
2 - MARGIN 2 VALUE............2
3 - MARGIN 3 VALUE............1
4 - MARGIN 4 VALUE............3

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

Type the number of the margin you want to alter to change the default setting. The areas on the printed page that correspond to each margin value are:

m1 = the number of lines from the top of the page down to, and including, the head title

m2 = the number of lines between the head title and the first line of text

m3 = the number of lines between the end of the text and the foot title

m4 = the number of lines at the bottom of the page, including the foot title

Alter the margin values according to your specific needs, then press **RETURN**. The maximum value for each margin is one less than the current page length value. The total values of all the margins plus the number of text lines on a page should equal your page length value.

If you do not use the head title (**.he**) or foot title (**.fo**) commands, 4 blank lines appear at the top and bottom of every printed page if you are using the default settings. Also, setting the first and fourth margin values to zero prevents the output of head or foot titles.

Return to the main FORMAT Configure menu by responding **N** (for no) to the prompt.

Paragraphs and Filling

Option 5 lets you change paragraph and filling modes permanently. Select **5** to see the options below:

```
1 - INITIAL FILL MODE.........YES
2 - INITIAL RIGHT ADJUST......NO
3 - PARAGRAPH INDENT..........+5
4 - PARAGRAPH SPACING.........1
5 - PARAGRAPH NEED VALUE......2

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

1 - INITIAL FILL MODE.....YES

When the fill mode is on, the FORMAT program gathers as many words entered at the PIE window as it needs to fill a specified line, so that words entered on different lines can be printed on the same line. The default is an undesirable setting when using FORMAT to print out computer programs, in which case you should change the default by responding N (for no).

2 - INITIAL RIGHT ADJUST.....NO

Initial right adjust inserts spaces between words or characters in a printed line so that the text is even, or justified, along the right-hand margin. If your printer is capable of incremental spacing, change this default by typing **Y** (for yes) to produce incrementally spaced and right-justified printed copy.

3 - PARAGRAPH INDENT.....+5

This setting affects how far the first line of your typical paragraph is indented. Any positive setting of up to 127 characters is legal. Enter the value you want and press **RETURN**.

4 - PARAGRAPH SPACING.....1

This option gives you the opportunity to change the number of blank lines inserted between paragraphs.

5 - PARAGRAPH NEED VALUE.....2

The **.pn** command includes an automatic **.ne** command that "looks ahead" to see how many text lines remain on the page. If the remaining lines are fewer than the lines "needed" for a paragraph, then a page break is forced before the new paragraph is begun. Enter whatever "need" you want invoked whenever a **.pp** or **.np** command that you have entered is encountered in a file, then press **RETURN**.

Return to the main FORMAT Configure menu after selecting appropriate paragraph and fill mode settings.

Program Disk Slot, Drive, Volume

The final option involves the use of more than one disk drive. Select **6**.

```
1 - SLOT.......................6
2 - DRIVE......................1
3 - VOLUME.....................0

DO YOU WANT TO CHANGE ONE
OF THE ABOVE VALUES?
```

Again, the slot, drive, and volume options are for users with more than one drive, specifying where the system diskette is to be found. The system diskette must be in the slot, drive, and volume specified whenever you transfer between PIE and FORMAT or access auxiliary files.

Set the SLOT to the slot of your disk controller card (usually 6), the DRIVE to the drive that contains the PIE Writer diskette, and the VOLUME, if using a hard disk system, to the necessary value.

Those are all the FORMAT Configure options. Change the appropriate settings to suit your typical needs, then answer **N** (for no) to the prompt:

```
DO YOU WANT TO LOOK AT OR MODIFY
ADDITIONAL VALUES IN ONE OF
THE ABOVE CATEGORIES?
```

Another prompt appears:

```
DO YOU WANT TO PERMANENTLY
UPDATE FORMAT?
```

Answer **Y** (for yes), and an updated copy of FORMAT is saved on the system diskette. The next time you boot the system, it will remember your preferred values. You can change FORMAT Configure defaults as often as you wish.

You can exit the System Menu and transfer to Applesoft BASIC by typing **5** in response to the System Menu.

Section 5, "Reference Section," reviews basic PIE and FORMAT text commands and introduces new, advanced commands. Some of the new commands perform sophisticated word processing functions. Explore the commands in the next section when you have mastered the commands and concepts already covered.

5

Reference Section

This section contains an explanation of the function of every PIE Text Editor, PIE Command Processor, and FORMAT Text Processor command in the PIE Writer system. Most of the basic commands have already been covered in earlier sections; however, "new," advanced commands are presented here for the first time. The Reference Card provides an abbreviated version of all the PIE Writer commands.

A Suggested Key Labels keyboard chart (p. 80); a reproduction of the Text Editor lesson screens (p. 98); an ASCII Reference Chart (p. 94); and a discussion of error messages you may encounter (p. 136) appear here as well.

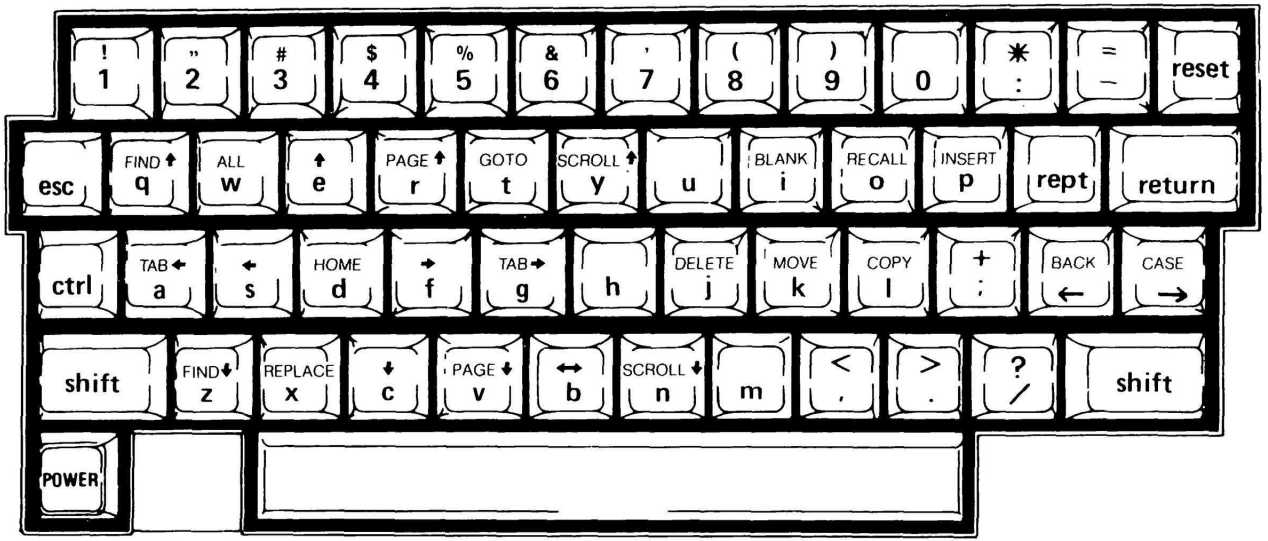
PIE TEXT EDITOR

PIE Text Editor commands are documented here in groups according to the similarity of their functions. Simple commands and their suggested key labels (a keyboard chart for which is reproduced on the next page) are displayed in simple command summaries under the heading "Command (Label)." Simple command functions appear to the right of the simple commands.

Some simple commands can be modified to form compound commands. All compound commands are initiated by typing the **ESC** key; some compound commands also take a modifier.

Modifiers are initiated with the **ESC** key and sometimes include a specific letter or the # character, an appropriate letter (c), an appropriate number (nn or mm), a string of characters (s1 or s2), or some combination of these. The modifiers, shown in regular type to the left of the simple commands, are prefixed to the simple commands to form compound commands. The compound command functions are described beneath the simple command summaries.

SUGGESTED KEY LABELS



CTRL-SHIFT-M — QUOTE
CTRL-SHIFT-N — REMOVE
CTRL-SHIFT-P — EXIT

Cursor and Screen Movement

The cursor indicates where your next typed character will appear. The cursor's column and line position are reflected in the Status Line at the bottom of the screen. With the following commands you can move the cursor to any position on the screen.

Modifier	Command (Label)	Simple Function
	CTRL-C (⇩)	Moves cursor down 1 line
	CTRL-E (⇧)	Moves cursor up 1 line
ESC nn	**CTRL-F** (⇨)	Moves cursor 1 space to the right
	CTRL-S (⇦)	Moves cursor 1 space to the left

Each of the simple commands trigger single-line scrolling when screen top and bottom limits are exceeded.

ESC nn CTRL-F moves the cursor to column 'nn.' Values from 1 to 64 are valid for the standard version {1 to 128 for 80-column versions}.

Modifier	Command (Label)	Simple Function
ESC S	**CTRL-G** (TAB ⇨)	Moves cursor 1 tab stop to the right
ESC C	**CTRL-G**	
ESC S	**CTRL-A** (TAB ⇦)	Moves cursor 1 tab stop to the left
ESC C	**CTRL-A**	
ESC 0	**CTRL-A**	
ESC nn	**CTRL-A**	

Attempting to tab beyond the confines of the screen has no effect, unless the Word Tab mode is on. In Word Tab mode, you can tab to the first character of the next word on the right or left: **CTRL-G** tabs to the word on the right, and **CTRL-A** tabs to the word on the left. If there are no more words on the line, you can tab to the first letter of the last word on the line above (**CTRL-A**) or to the first letter of the first word on the line below (**CTRL-G**).

ESC S CTRL-G sets the bell column at whatever column the cursor is in; but **ESC C CTRL-G** clears the bell column, regardless of the cursor position and the column in which the bell is set and whether or not you are in Word Tab mode. The bell column is indicated by a reverse video character at the top of the screen.

In column tabbing, **ESC S CTRL-A** sets a tab at the column the cursor is in; **ESC C CTRL-A** clears a tab at the cursor; **ESC 0 CTRL-A** clears all tabs, regardless of the cursor position and the number of columns in which the tabs are set; and **ESC nn CTRL-A** sets a tab every 'nn' columns beginning at the first column, a valid value for 'nn' being from 1 to 64 {1 to 128 for 80-column}.

Modifier	Command (Label)	Simple Function
	CTRL-D* (HOME)	Moves cursor to the top left or bottom left corner of screen
ESC	**CTRL-B*** (⇔)	Moves cursor to beginning or end of line

CTRL-D (* means "toggle") alternates between one of two functions: it toggles back and forth between moving the cursor to the top left-hand corner and moving it to the bottom left-hand corner of the screen. From any other cursor position, the **CTRL-D** moves first to the top left-hand corner.

CTRL-B initially moves the cursor to the end of whatever line it is on, then toggles back and forth between moving the cursor to the beginning and to the end of that line.

ESC CTRL-B has a single function: it deletes all the blank lines at and beneath the cursor.

Modifier	Command (Label)	Simple Function
ESC nn	**CTRL-V** (PAGE ⇩)	Advances screen 21 lines (1 page)
ESC	**CTRL-V**	
ESC nn	**CTRL-R** (PAGE ⇧)	Reverses screen 21 lines (1 page)
ESC nn	**CTRL-N** (SCROLL ⇩)	Advances cursor and screen 1 line
ESC	**CTRL-N**	

Modifier	Command (Label)	Simple Function
ESC nn	**CTRL-Y** (SCROLL ⇧)	Reverses cursor and screen 1 line
ESC	**CTRL-Y**	

CTRL-V advances the screen 21 lines, or 1 "page," and **CTRL-R** reverses the screen 21 lines. The cursor remains in its original position on the screen.

ESC nn CTRL-V advances the screen 'nn' pages; **ESC nn CTRL-R** reverses the screen 'nn' pages.

ESC CTRL-V moves the line the cursor is in to the top of the screen, homing the line. The cursor remains in the same column.

CTRL-N advances the screen 1 line. The cursor advances with the line it was on. **CTRL-Y** reverses the screen and cursor 1 line.

ESC nn CTRL-N advances the screen and cursor 'nn' lines, and **ESC nn CTRL-Y** reverses the screen and cursor 'nn' lines. **ESC CTRL-N** advances the screen and cursor $^1/_2$ page, and **ESC CTRL-Y** reverses the screen and cursor $^1/_2$ page.

Modifier	Command (Label)	Simple Function
ESC	**CTRL-T** (GOTO)	Moves cursor to file beginning
ESC nn	**CTRL-T**	

CTRL-T moves the cursor and screen to the beginning of the file you are working on; **ESC CTRL-T** moves the cursor and screen to the end of that file. **ESC nn CTRL-T** moves the cursor and screen to line 'nn.'

While PIE Writer is carrying out commands that take a few seconds to complete, such as the **CTRL-T** command, you can continue entering commands (or text). This is called "type-ahead." The computer remembers what you have typed and enters it at the cursor's original position after your command has been carried out.

Inserting and Deleting

The cursor and screen movement commands leave text in their path intact. Destructive cursor movements are necessary for basic editing functions.

Modifier	Command (Label)	Simple Function
ESC s1	**CTRL-P*** (INSERT)	Begins or ends insertion of characters at cursor
	◀ (BACK)	Deletes character to the left and backspaces
ESC nn	**CTRL-I** (BLANK)	Inserts a blank line at cursor
ESC	**CTRL-I**	
ESC c	**CTRL-J** (DELETE)	Deletes character at cursor and moves other characters on line 1 space to the left
ESC	**CTRL-J**	
ESC	**CTRL-SHIFT-N** (REMOVE)	Deletes word at cursor
ESC nn	**CTRL-SHIFT-N**	
ESC # mm,nn	**CTRL-SHIFT-N**	

CTRL-P toggles back and forth between initiating the mode by which a character or string of characters can be inserted at the cursor (INSERT appears on the Status Line) and ending that insertion mode. If there is not enough space on the line for character insertion, the error message NOT ENOUGH ROOM ON LINE is displayed on the Status Line.

ESC s1 CTRL-P allows you to insert 's1'—a string of characters—at the cursor, forcing all other text on the line to make room by moving to the right. **ESC CTRL-P** inserts 's1' at the cursor again. The string, a word or words, is limited to 26 {or 58} characters.

The ◀ command deletes text as it backspaces. This command has no effect if the cursor is in column 1. In Insert mode, initiated by **CTRL-P**, the ◀ can be used to drag every character to the right of the cursor to the left.

CTRL-I inserts a blank line, and **ESC nn CTRL-I** inserts 'nn' blank lines, at the cursor.

An **ESC CTRL-I** splits a line at the cursor and moves whatever text was to the right of the cursor to the left margin of the new line.

The **CTRL-J** command deletes a character at the cursor and shifts all the text to the right of the cursor to the left; **ESC c CTRL-J** deletes all the

characters on a line from the cursor to the first occurrence of 'c' on the same line, any character or letter being a valid 'c', and the text to the right of the deletion shifts left; and **ESC CTRL-J** deletes every character to the end of a line from the cursor.

CTRL-SHIFT-N deletes an entire word regardless of the cursor's position within that word. The text to the right of the cursor shifts left. If the cursor is on the space preceding a word, all spaces to the left of the word are deleted. In PPWrap mode, if the word deleted is the only word on a line, a line delete is performed, closing up the created space between the lines above and below.

ESC CTRL-SHIFT-N deletes an entire line regardless of the cursor's position within the line, and all subsequent lines are moved up. **ESC nn CTRL-SHIFT-N** deletes 'nn' lines at and beneath the cursor.

ESC # mm,nn CTRL-SHIFT-N deletes lines 'mm' to 'nn,' including lines 'mm' and 'nn.' The nature of this command requires the loading of code from the PIE AUX file, so the Status Line will show BUSY while the command is being carried out.

A '$' may be typed as 'nn' to represent the last line n the file.

Copying and Moving Lines

The ability to copy lines into memory and move them to different positions in your text is a powerful editing function. Using the copy and **CTRL-T** (GOTO) functions, any amount of text may be copied from one position to another position within a file. Additionally, the information held in the copy "buffer" will remain after loading in a new file so that short blocks of text may be moved between files easily.

Modifier	Command (Label)	Simple Function
ESC nn	**CTRL-L** (COPY)	Copies line at cursor into memory
ESC nn	**CTRL-K** (MOVE)	Copies line at cursor into memory and deletes it from screen
ESC	**CTRL-K**	
ESC # mm,nn	**CTRL-O (RECALL)**	Retrieves line put in memory with a COPY or MOVE and inserts it at cursor

CTRL-L copies a single line of text into memory. The Status Line blinks once while the line at the cursor is being copied. **ESC nn CTRL-L** copies 'nn' lines beginning at the cursor into memory, up to a maximum of 21 lines of text, or 1 full window.

The **CTRL-K** command copies a single line of text at the cursor into memory and deletes it from the screen. This is the safest method you can use for deleting a line of text. You can save a copy of up to 1 window, or 21 lines, of text with **ESC nn CTRL-K**, which saves 'nn' lines in memory while deleting them from the screen.

ESC CTRL-K joins a line below the cursor to the end of the line the cursor is on. The cursor must be positioned following the final letter in a line, or else the Status Line says to CLEAR TO EOL (end of line) FIRST. If the line below the cursor is too long to be appended to the current line, a NOT ENOUGH ROOM ON LINE message appears on the Status Line.

The **CTRL-O** command allows you to recall the set of lines placed in memory with either **CTRL-L** (COPY) or **CTRL-K** (MOVE). The text recalled is inserted at the current cursor position. The cursor may be repositioned and the **CTRL-O** command may be used again to insert as many copies of the block as you like.

ESC # mm,nn CTRL-O moves a range of any number of lines. Move the cursor to the place in the text where you want to move lines 'mm' to 'nn.' Enter **ESC # mm,nn** command, and the range of lines 'mm' to 'nn' will be retrieved from memory and placed at the cursor. The system diskette with the PIE AUX file must be in the system drive to execute this command.

Again, '$' may be used to represent the end of file.

Search and Replace

A "string" is a sequence of characters. A string might be a sentence, a word, or a single character. PIE limits search strings to a maximum of 26 {58} characters. PIE can be instructed to search for a string and to position the cursor at the string when it has been found. Failed searches produce a SEARCH KEY NOT FOUND message in the Status Line. The word "KEY" in the message refers to the search string.

Modifier	Command (Label)	Simple Function
ESC s1	**CTRL-Z** (FIND ⇩)	Searches forward for 's1' again
ESC s1	**CTRL-Q** (FIND ⇧)	Searches backward for 's1' again

To find a string, type **ESC** to display the ARG: prompt on the Status Line. Enter the string (s1) you want found. When the string is correctly spelled, press **CTRL-Z** to search toward the end of the file and press **CTRL-Q** to search toward the beginning of the file. PIE Writer remembers your search string until you type in a new one. To search for the same string again, press **CTRL-Z** to search forward again, and press **CTRL-Q** to search backward again.

If the string exists but is split across two lines it will not be found.

A search string can contain ambiguous or "wild card" characters (except for the first character). A wild card character matches any letter. DEL is the wild card character, which is entered by typing ➡ (CASE) **3** or ➡ (CASE) #. (See "Entering Control and Nonkeyboard Characters" on p. 93 for a full explanation on how to enter nonkeyboard characters.)

For instance, a string consisting of the letter "C," the letter "A," and the DEL character will match "CAT," "CAP," and "CAR" during a search.

Modifier	Command (Label)	Simple Function
ESC s1		
ESC s2	**CTRL-X** (REPLACE)	Replaces 's1' on screen with 's2' again
ESC s2	**CTRL-X**	
ESC s1		
ESC s2	**CTRL-W** (ALL) **CTRL-X**	

The two strings (s1 and s2) refer to the search string and replacement string, respectively. As can be seen by the many command options above, there are several sequences in which these two strings may be entered.

ESC s1 ESC s2 CTRL-X searches forward for a string (s1) and replaces it with another string (s2). Simply press **CTRL-Q** or **CTRL-Z** to find the search string in another part of the file. Typing **CTRL-X** performs another replacement at the next occurrence of the search string. To enter a different replacement string but leave the search string intact, use **ESC s2 CTRL-X**.

If the replacement string (s2) causes the line to exceed the available space on the line, the command will not execute, and the Status Line will display NOT ENOUGH ROOM ON LINE. If the search string cannot be found, the replacement cannot be made, so the Status Line displays SEARCH KEY NOT FOUND.Remember that **CTRL-X** does only a forward search.

To change every occurrence of a string throughout the text, use **ESC s1 ESC s2 CTRL-W CTRL-X**. This looks for the search string (s1) and replaces it with the replacement string (s2) everywhere in the file. This function is often called a "global" search and replace.

Every occurrence of the search string will be replaced, unless a line is made too long by the replacement string, in which case the command quits. You may edit the line, then press **CTRL-W CTRL-X** again, and the global search and replace continues forward searching and replacing using the same strings.

The sequence **CTRL-W CTRL-X** takes the most recent search and replacement strings and performs a global search and replace from the cursor position to the end of the file.

Note that using **ESC s1 CTRL-P** to make an insertion changes the current replacement string.

Text Entry Modes and Help

PIE Writer allows you to select different text entry modes. Also, you can access a list of all the simple commands and their functions if you have loaded the Help file previously.

Modifier	Command (Label)	Simple Function
ESC	**RETURN**	Carriage return
ESC I		
ESC W*		
ESC H		

The standard typewriter requires a carriage return at the conclusion of every typed line. PIE Writer's Manual mode works the same way, the **RETURN** key executing a typical carriage return. The Manual mode is the one you are in when you first enter the Text Editor.

ESC RETURN (or **ESC P RETURN**) sets the PPWrap mode from the default Manual mode. PPWrap mode allows you to type continuously without pressing the **RETURN** key: whenever a character is typed beyond the bell column, all the previous characters in the same word are automatically moved down to the next line.The entire word wraps around the screen and begins flush left on the next line below. When the text wraps around or when the **RETURN** key is pressed, any text beneath the line being typed shifts down at the end of a line on the screen to make room for the newly entered text.

ESC I RETURN lets you enter the Indent mode. In this mode, pressing the **RETURN** key moves the cursor to the position below the leftmost character on the previous line. This is useful for entering outlines, lists, and program text, such as Pascal or Assembly code.

In the Indent mode <**RETURN**>
a line indented this much <**RETURN**>
is indented again automatically <**RETURN**>
after every carriage return.

You must press **RETURN** to perform a carriage return in the Indent mode. Text does not wrap around the screen automatically. Pressing **RETURN** does make room for any new text you may want to enter, however, by forcing all text below the cursor down one line.

ESC RETURN from either PPWrap or Indent mode returns you to Manual mode.

Word Tabbing is a powerful variation on typewriter tabbing, allowing the user to tab to the next or previous word in a line with a **CTRL-G** or **CTRL-A** command.

The type of tabbing (word or column) is set separately for each of the three text entry modes. Initial default settings are Word Tabbing for PPWrap mode and column tabbing for Manual and Indent modes. **ESC W RETURN** toggles the tabbing method for the current text mode. If you shift between modes, the type of tabbing previously selected in that mode is restored.

ESC H RETURN displays the Help screen, a listing of all the simple functions and their command keys. However, this command is available only if the Help file already has been loaded from the Command Processor by typing **%H**. (See page 114.)

Using PIE with the Help screen loaded reduces the size of your edit buffer by 1024 characters. It is intended to be used while learning the Text Editor commands.

By pressing **RETURN** while viewing the Help screen, you are returned to the precise point in the file where you left off.

Displaying Upper- and Lowercase Letters

The unmodified Apple II can only display uppercase letters on the screen. It differentiates between capital and lowercase letters by showing black characters on a white background (reverse video) for uppercase and white characters on a black background for lowercase letters. This way, the relatively few capital letters stand out from the majority of lowercase letters in normal text entry.

Modifier	Command (Label)	Simple Function
➡	➡ (CASE)	Makes next character entered uppercase

Using the ➡ key lets you enter only the next character in uppercase reverse video. Pressing the ➡ twice locks in uppercase entry, so that you can type in strings of capital letters without continually pressing the ➡ prior to each character typed. Pressing the ➡ twice more returns you to lowercase entry mode.

Whenever PIE Writer is in the lowercase mode, the Status Line displays LOCASE. Absence of LOCASE on the Status Line indicates that you are in uppercase entry mode.

Two methods are available for displaying true upper- and lowercase letters. Adapters are sold by several manufacturers that allow upper- and lowercase display on standard Apple II systems. A PIE Configure option (see p. 000) notifies PIE Writer that you have added this feature to your system for true character display.

Also, 80-column display boards include true upper- and lowercase display capabilities.

On p. 000, a simple hardware modification is described that makes the **SHIFT** key perform the way it does on a regular typewriter.

Contact your dealer for additional information on lowercase adapters, 80-column display boards, and the **SHIFT** key modification.

Cursor-Defined Commands

With cursor-defined commands, you allow the horizontal and vertical motion of the cursor to define the area of the screen you want a command to be effective in, so that the same command can be applied to more than one line of text at a time. Cursor-defined commands can be carried out only for on-screen text below or to the right of the cursor's position before the cursor-defined command is entered.

Cursor-defined commands allow you to count lines and columns, using the cursor, to specify the range of a command, just as a numeric value can be used to count those same lines and columns. For example, **ESC 8 CTRL-I** inserts 8 blank lines at the cursor.

The cursor-defined analog of this command is described below.

Modifier	Command (Label)
ESC @⇨⇩	**CTRL-I** (BLANK)
ESC @ ⇩	**CTRL-L** (COPY)

Modifier	Command (Label)
ESC @ ⇩	**CTRL-K** (MOVE)
ESC @⇨⇩	
ESC @ ⇩	**CTRL-SHIFT-N** (REMOVE)
ESC @⇨⇩	

The ESC @ notation, followed by a ⇩, indicates that the modifier for that particular cursor-defined command can only be a simple vertical command for that cursor-defined function. The ESC @, followed by a ⇨⇩, indicates that both vertical and horizontal simple cursor movement commands can be used as modifiers.

The use of the @ symbol in the above notation does not indicate that the @ key is used to enter these commands. Cursor-defined modifiers are entered by first typing **ESC**, then entering one of the seven simple cursor movement commands noted below. (As soon as a cursor command is typed following **ESC**, the original cursor position is replaced on the screen with the @ symbol, and the message CURSOR DEFINED is displayed on the Status Line.) The @ symbol in the notation simply represents the starting point of the cursor's movement.The arrows indicate the direction of the legal net cursor movement.

The **CTRL-C**, **CTRL-E**, and **CTRL-D** simple commands are capable of functioning vertically only. Net cursor movement must be below the original cursor position. **CTRL-D**, as a cursor move, works differently. Instead of toggling between Home and Bottom Home, as a modifier it toggles between the original cursor line and the bottom line on the screen, remaining in the current column.

The **CTRL-F**, **CTRL-S**, **CTRL-G**, and **CTRL-A** simple commands are capable of functioning horizontally and repeating that function vertically with a **CTRL-C**, **CTRL-E**, or **CTRL-D**. Net cursor movement is to the right of the original cursor position.

ESC @⇨⇩ **CTRL-I** performs a cursor-defined block move right. Assume that the cursor is positioned on the 't' in the word 'this' below, and you want to move two vertical lines beginning in this column to the right 8 spaces.

```
THIS TEXT IS HERE
HERE'S SOME MORE
AND YET MORE
```

Press **ESC** and then press **CTRL-G**. The Status Line states CURSOR DEFINED.

The cursor's position before you entered the command is marked with a @ on the screen.

```
@HIS TEXT IS HERE
HERE'S SOME MORE
AND YET MORE
```

Assuming your tab stops are set to 8, the **ESC CTRL-G** tabs the cursor to the second 't' in 'text':

```
@HIS TEX[T] IS HERE
HERE'S SOME MORE
AND YET MORE
```

CTRL-C moves the cursor down one line so that the cursor is positioned over the 'o' in 'some' on the second line:

```
@HIS TEXT IS HERE
HERE'S S[O]ME MORE
AND YET MORE
```

CTRL-I inserts spaces from @ to the current cursor position:

```
        THIS TEXT IS HERE
        HERE'S SOME MORE
AND YET MORE
```

In the above sequence, if **CTRL-D** were substituted for **CTRL-C**, every line from the original cursor position to the bottom of the screen would be indented 8 spaces.

ESC @⇨⇩ **CTRL-K** performs a cursor-defined block move left. The **CTRL-K** command is preceded by a simple horizontal command (to indicate the number of spaces you want to shift the text left) and a simple vertical command (to indicate how many lines you want shifted). Note that the cursor movement is in the direction opposite that of the text shift. In the example below, you can shift the block of text to the left by first positioning the cursor in the column where you want the first letter of the line to end up, then pressing **ESC**:

```
@       THIS TEXT IS HERE
        HERE'S SOME MORE
        AND YET MORE
```

Move the cursor so that it is positioned over any part of the text in the first line with **CTRL-F** or **CTRL-G** commands, then use a **CTRL-C** to move the cursor anywhere on the line below it. Type a **CTRL-K** command to see the screen below:

```
THIS TEXT IS HERE
HERE'S SOME MORE
     AND YET MORE
```

ESC @⇨⇩ CTRL-SHIFT-N shifts text to the left, but also deletes characters as defined by the cursor movement. In the example above, the cursor must have been positioned horizontally on the 't' in 'this,' then moved to the 'h' in 'here's' with a **CTRL-C** to get the same result. Positioning the cursor anywhere within the lines would have deleted all the characters to the left of the cursor, then shifted the remaining characters on the lines left to the column where the asterisk was first set.

ESC @⇩ CTRL-L places lines into memory as defined by the cursor movement. The entire lines are placed in memory regardless of the column the cursor is in.

ESC @⇩ CTRL-K places lines into memory as defined by the cursor movement and deletes them from the screen.

ESC @⇩ CTRL-SHIFT-N deletes lines as defined by the cursor movement, and the lines below those deleted close up the space created.

Entering Control and Nonkeyboard Characters

Control characters cannot be entered into a text file directly using the Text Editor for the simple reason that all of the keyboard control characters are interpreted as Text Editor commands. Naturally, it is desirable to be able to enter any ASCII character into your file. (A typical application would be to imbed printer commands in your file.)

Modifier	Command (Label)	Function
	CTRL-SHIFT-M (QUOTE)	Allows entry of next character as a control code

CTRL-SHIFT-M lets you enter any control character literally, that is, without interpretation. Executing a **CTRL-SHIFT-M** produces the message ENTER CTRL CHAR on the Status Line.

For control characters in the range CTRL-@ to CTRL-Z, simply type the corresponding key; you do not need to press CTRL. For control characters not available on the Apple keyboard, enter the appropriate translation charac-

ter. (Refer to the ASCII Reference Chart on pp. 94–97 for information on entering control characters.) The control character you enter flashes on your screen. {On 80-column versions, it depends on your board, but they are usually displayed in inverse type.}

Most printing characters are entered simply by typing the keys (the ➡ key is normally used to toggle between uppercase and lowercase, but the **SHIFT** key can be used if the hardware modification has been made).

Keys unavailable on the Apple keyboard can be entered using the ➡ key followed by the non-alpha key listed under "Translation Character" in the ASCII Reference Chart. For machines without lowercase adapters or 80-column boards, these will be displayed by indicated, different characters.

Modifier	Command (Label	Function
	➡ (CASE) 'n'	Allows entry of non-keyboard characters by translating 'n'

The special characters, ^, [, and @, can only be entered using the **SHIFT** key when in uppercase lock mode (uppercase lock corresponds to the standard Apple keyboard).These may be entered from the lowercase mode using the translation characters.

Again, the ASCII Reference Chart below indicates the required keystrokes necessary to enter these nonkeyboard characters.

ASCII CHARACTER CODES

Control Characters

Char	Dec	Hex	Translation
NUL	0	00	
SOH	1	01	
STX	2	02	
ETX	3	03	
EOT	4	04	
ENQ	5	05	
ACK	6	06	
BEL	7	07	
BS	8	08	

Char	Dec	Hex	Translation
HT	9	09	
LF	10	0A	
VT	11	0B	
FF	12	0C	
CR	13	0D	
SO	14	0E	
SI	15	0F	
DLE	16	10	
DC1	17	11	
DC2	18	12	
DC3	19	13	
DC4	20	14	
NAK	21	15	
SYN	22	16	
ETB	23	17	
CAN	24	18	
EM	25	19	
SUB	26	1A	
ESC	27	1B	;
FS	28	1C	<
GS	29	1D	=
RS	30	1E	>
US	31	1F	?

Use shift key to enter "Upper Case" characters (translation type) after CTRL-SHIFT-M.
i.e. CTRL-SHIFT-M
SHIFT(TRANSLATION CHAR.)

Printing Characters

SP	32	20	
!	33	21	
"	34	22	
#	35	23	
$	36	24	
%	37	25	
&	38	26	
'	39	27	
(	40	28	
)	41	29	
*	42	2A	
+	43	2B	
,	44	2C	
-	45	2D	
.	46	2E	
/	47	2F	

when entering a sequence of commands such as ESC T [CHR$(27);CHR$(84)] only use CTRL-SHIFT-M for the control character (ESC) then just type the next character "T"

Char	Dec	Hex	Translation
0	48	30	
1	49	31	
2	50	32	
3	51	33	
4	52	34	
5	53	35	
6	54	36	
7	55	37	
8	56	38	
9	57	39	
:	58	3A	
;	59	3B	
<	60	3C	
=	61	3D	
>	62	3E	
?	63	3F	
@	64	40	
A	65	41	
B	66	42	
C	67	43	
D	68	44	
E	69	45	
F	70	46	
G	71	47	
H	72	48	
I	73	49	
J	74	4A	
K	75	4B	
L	76	4C	
M	77	4D	
N	78	4E	
O	79	4F	
P	80	50	
Q	81	51	
R	82	52	
S	83	53	
T	84	54	
U	85	55	
V	86	56	
W	87	57	
X	88	58	

Char	Dec	Hex	Translation
Y	89	59	
Z	90	5A	
[	91	5B	<
\	92	5C	/
]	93	5D	>
^	94	5E	"
_	95	5F	=
`	96	60	
a	97	61	
b	98	62	
c	99	63	
d	100	64	
e	101	65	
f	102	66	
g	103	67	
h	104	68	
i	105	69	
j	106	6A	
k	107	6B	
l	108	6C	
m	109	6D	
n	110	6E	
o	111	6F	
p	112	6G	
q	113	6H	
r	114	72	
s	115	73	
t	116	74	
u	117	75	
v	118	76	
w	119	77	
x	120	78	
y	121	79	
z	122	7A	
{	123	7B	(
|	124	7C	!
}	125	7D	)
~	126	7E	+
DEL	127	7F	#

Exiting the Editor and Transferring to the Command Processor

The final Text Editor command transfers you to the Command Processor (CP).

Modifier	Command (Label)	Function
	CTRL-SHIFT-P (EXIT)	Exits Editor and returns to Command Processor

You exit all PIE Text Editor files and transfer to the PIE Command Processor by executing a **CTRL-SHIFT-P**. Your text remains intact in the buffer. You can reedit it, but it will not be stored permanently on diskette until you enter a save or write command while in the Command Processor.

Text Editor Lesson Screens

The screens for the eighteen Text Editor lessons are reproduced below simply to provide a hard copy of what they look like when they are taken in sync. Make sure when you take the lessons that the underscore is always in the top right-hand corner of the screen before scrolling on to the next exercise.

Lesson 1

```
_<---- This is the cursor ---<

The cursor shows exactly where
your next typed character will
appear.

You will be taught to move the
cursor all around the screen to
write or make changes at the
cursor much like you would with
a pencil and paper.

Now scroll to the next screen
for some practice moving the
cursor.

Find the "CTRL" key.  While
holding the CTRL key down,
press the letter "V" (as in
Victory) once for the
CTRL-V (PAGE DOWN) function.
```

Lesson 2 _<----This is the "Home" position.

The CTRL-D key sequence moves you back and forth between Home and Bottom Home. Try it. Hold CTRL down and tap "D" a few times.

Any time you want to go to either "home" from anywhere on the screen, you can use CTRL-D (HOME).

Throughout the PIE Tutorial, make sure the underscore and the cursor are in the Home position before scrolling on to the next screen. This keeps lesson and screen "in sync".

Home the cursor and type CTRL-V.

_<----This is "Bottom Home."

Lesson 3 _ The "Lopsided Cross"

Surrounding the D key are four keys that, when pressed with CTRL, move the cursor horizontally and vertically around the screen. They form sort of a lopsided cross on your keyboard.

```
CTRL-E = up          -------
                    !  E    !
CTRL-C = down       !       !
                    ! S D F !
CTRL-S = left       !       !
                    !    C  !
CTRL-F = right       -------
```

Scroll to the next screen and try these moves. Make sure the cursor is at Home with CTRL-D, then press CTRL-V once, and scroll on.

Lesson 4 _ Cursor Control Practice

Use the "lopsided cross" keys to move the cursor. try to put the cursor on the "X" in the box.

If you go too far to the right, the screen will shift. Press RETURN to get back to the lesson.

CTRL-E = up

CTRL-C = down

CTRL-S = left

CTRL-F = right

```
---
!   !
! X !
!   !
---
```

When you're ready to roll, Home the cursor (use the keys above or CTRL-D), then press CTRL-V.

Lesson 5 _ THE STATUS LINE . . .

is the information line beneath this window. It lists cursor position and other information.

If you want to know what column the cursor is in or what line it is on, check the Status Line.

Column number is at the lower left, line number on the lower right.

The info in the Status Line changes from time to time, depending on what you are doing. But it is always up-to-date.

Make sure the cursor is Home, then press CTRL-V, and scroll on.

Lesson 6 _ Beyond the right-hand margin

Do you see some >'s running down the right-hand margin of this page?

MYSTERY LINES >---------->

Each > tells you there is text in the window BEYOND that margin.

To see the end of the mystery lines:

1. Use CTRL-F to move the cursor to the right. Watch the Status Line's COL number change. When you reach COL 38, STOP!

2. Now press CTRL-F one more time. Do you understand what happened?

Press CTRL-D, CTRL-V to continue.

Lesson 7

```
^       ^       ^       ^       ^
tab     tab     tab     tab     tab
```

Note the tab markers (^) at the top of the window.

Tabbing is a quick way to move across the page. To tab to the right, press CTRL-G (TAB >). Notice that you stop at a tab marker each time.

You can also tab to the left. Press CTRL-A (TAB <) the same number of times. You will stop at the same tabs on the way back.

When you are ready to move on, make sure the cursor is home (CTRL-D), and scroll on (CTRL-V) to the next lesson.

Lesson 8 _ CTRL-B -- the APPENDING key

Pressing CTRL-B (<-->) moves the cursor to the end of whatever line you are on. This makes it easy to append words to the ends of phrases. Press CTRL-B to return to the left margin again.

Position the cursor at the beginning of each line below. Then use CTRL-B to append something to each line.

---

1. Dear Dr. Watson,

2. Please send me

3. If only I hadn't

Press CTRL-D & CTRL-V to move on.

Lesson 9 _ Scrolling pages and lines

You know that CTRL-V (PAGE DOWN) scrolls you down a page. Try scrolling up a page with CTRL-R (PAGE UP), then return here.

There are other magical scrolling moves. Try scrolling up a line with CTRL-Y (SCROLL UP). The top line will scroll off the screen. Then scroll down a line with CTRL-N (SCROLL DOWN) and bring the top line back. The manual describes more scrolling maneuvers later on.

You will be back in the world of paper pages soon enough. In the meantime, practice your new moves. Then, be sure the page is properly aligned, and type CTRL-V.

Lesson 10 _ UPPERCASE; lowercase

The standard Apple screen does not display lowercase letters. You can type upper- AND lowercase on the keyboard, but the screen will display only uppercase. Lowercase letters will not appear until you print your document, but you can still tell which is which:

UPPERCASE = Black letters in white rectangles

lowercase = White letters on plain black screen

The next lesson will teach you how to enter upper- and lowercase text. Make sure the page is aligned (CTRL-D), then press CTRL-V.

Lesson 11 _ The --> (CASE) key

To shift from lowercase (lc) letter entry to UPPERCASE (UC), use the --> (CASE) button at the far right on your keyboard.

1. To type 1 letter in UC, press --> once, then type the letter. The word "SHIFT" will appear on the Status Line. You will be returned to lc automatically.

2. You can lock UC in by pressing --> twice, then enter as many UC letters as you want. Return to lc by pressing --> twice more. "LOCASE" will appear on the Status Line.

Press CTRL-D, CTRL-V for Lesson 12.

Lesson 12

_ type something on the screen, using the --> (CASE) key. Check the Status Line response to each shift. You may type over any of the words in the typing area below:

*** typing area ***

If you go beyond col 38,
use the RETURN key.

To type 1 letter in uc: -->

To lock in UC: --> -->

To unlock UC for lc again: --> -->

To scroll on, use CTRL-D & CTRL-V.

Lesson 13

_ The SHIFT key says, "%$&*?!"

The SHIFT key does not make upper-case letters as it does on a typewriter. It gives you access to the upper register characters instead. Whether you have been entering upper- or lowercase letters on the screen, hold the SHIFT key down while typing the upper register character needed. Move the cursor to the area below, and practice typing upper register characters.

*** typing area ***

To scroll on, press CTRL-D & CTRL-V.

Lesson 14 _ Can you delete this line?
How about this one?

"DESTRUCTIVE" CURSOR MOVES

The space bar and the <- (BACK) key move the cursor and delete characters at the same time.

The space bar moves the cursor 1 space to the right and deletes as it continues moving. the <-- (BACK) key works like an erasing backspace, moving the cursor and deleting the character 1 space to the left.

Move the cursor to the top of the screen and delete the two lines between the arrows. Then press CTRL-D and CTRL-V to learn another simple editing command.

Lesson 15 _ CTRL-J erases mistakes

CTRL-J (DELETE) eats mistakes whole, without leaving a trace. It makes fixing mistakes as easy as making them in the first place.

Move the cursor until it is directly on a flashing letter in a word below, then press CTRL-J to erase the flashing letter:

typoDs winDoWs misGnomers

Now position the cursor on the first letter in the next line and keep pressing CTRL-J to reveal a secret.

Gobble this line for a secret message.

Press CTRL-D, CTRL-V, and scroll on.

Lesson 16 _ CTRL-P for insert magic

When you need to insert characters within a line, CTRL-P (INSERT) automatically creates space for as many characters as you want to type. Each character at and to the right of the cursor moves one space to the right.

Position the cursor directly on the letter "s" in "puse" below, then press CTRL-P. Note that the Status Line will say "INSERT."

The patient had no puse.

Now type the lowercase letter "l." Press CTRL-P to leave the insert mode, then CTRL-D and CTRL-V to learn more about CTRL-P (INSERT).

Lesson 17 _ Shifting lines with CTRL-P

With CTRL-P (INSERT), you can move a line to the right by inserting spaces in front of it. Position the cursor so that it flashes on the first "l" in the top line below. Press CTRL-P, then tap the space bar repeatedly until the two lines are even.

```
line this line up
                    with this line
```

In the insert mode, you can also use the <-- (BACK) key to shift lines to the left. With the cursor just to the left of the top line, press <-- repeatedly to realign the phrase along the left margin. Now press Ctrl-P, CTRL-D, & CTRL-V for more.

Lesson 18 _ EXIT with CTRL-SHIFT-P

If you wish to take the PIE Tutorial again, press CTRL-T (GOTO), which takes you the beginning of any PIE Writer file. You can review specific lessons by scrolling the window backward and forward a line or a pae at a time with CTRL-R (PAGE UP), CTRL-Y (SCROLL UP), CTRL-V (PAGE DOWN), and CTRL-N (SCROLL DOWN).

You exit the PIE Tutorial the same way you exit every PIE Writer file, with a CTRL-SHIFT-P (EXIT). After you exit, you will be transferred to the Command Processor.

To exit, press CTRL-SHIFT-P (EXIT) all at the same time.

PIE COMMAND PROCESSOR

You are transferred to the Command Processor after choosing the first option in the System Menu, PIE TEXT EDITOR, after exiting the PIE Text Editor with a **CTRL-SHIFT-P** command, or returning to PIE from FORMAT. The Command Processor handles the manipulation of all PIE Writer files on diskette and provides statistics and information on the status of current files.

The Command Processor has three types of commands: Control, Input/Output, and Auxiliary.

All commands must be followed by a **RETURN** and may be given only in response to the Command Processor prompt:

```
COMMAND?:
```

Prior to pressing **RETURN**, you may delete an entry by typing **CTRL-X**. For users with the Autostart ROM, the Command Processor also supports cursor moves and screen copying using **ESC I-J-K-M** as described in the Apple II Reference Manual.

Control Commands

CP Command	Function
N(EW)	Edit new file; clear current file
E(DIT) n	Reedit current file (optionally starting at line 'n')
F(ORMAT)	Transfer to FORMAT program
C(ATALOG)	Catalogs current drive
?	Display remembered filename (fn), set with L fn, S fn, < fn, > fn
LE(NGTH)	Display remembered filename, length, and available memory remaining
CTRL-Y	Enter Apple system monitor; **CTRL-Y** returns you to Command Precessor
CALL addr	Calls machine language routine at addr; may be signed decimal or $hex

N (or **NEW**)—Clears the edit buffer before beginning a new text file.The **N(EW)** command is followed by the precautionary prompt OK TO CLEAR?: A **Y** (for yes) response produces the empty text window; a **N** (for No) redisplays the CP prompt.

E (or **EDIT**) **n**—Returns you to the text editor without clearing the previous contents of the buffer, so that you can begin or resume editing your file. This command must be used after loading a file from diskette for reediting.

The cursor appears at the beginning of your text for a recently loaded file; otherwise, the cursor reappears at its previous position if you have worked with this file and have transferred to the Command Processor temporarily (for example, to update your diskette file).

Entering an optional 'n' value allows you to specify the line number of the file you would like the cursor to appear at the beginning of. If the line number is greater than the highest line number in the text buffer file, the cursor appears at the beginning of the file's last line. (The command **E $** also jumps to the end of the file.)

F (or **FORMAT**)—Transfers you to the FORMAT Text Processor, which formats your file for output to the printer, the screen, or diskette.

C (or **CATALOG**)—Displays the filenames on the currently active diskette.You may specify the slot, drive, and volume number of the disk drive whose catalog you want displayed if you are using more than one drive.

?—Displays the title of the last loaded or saved file. This filename becomes the 'remembered filename' and may be used as a shorthand notation for that filename in any command, as in the following examples:

- **S?**—*Saves remebered 'B' type file to diskette (you can optionally specify a different slot, drive,and volume number).*
- **S?.BAK**—*Saves a file with the characters .BAK appended to the remembered filename.*
- **>?**—*Outputs 'T' type file.*
- **L?**—*Loads file.*

Using the ? token as part of a filename when appending or saving to a 'side file' (as in S ?.BAK) does not change the remembered filename.

LE (or **LENGTH**)—Displays the remembered filename, the number of characters in the file, and the remaining memory available.

CTRL-Y—Transfers you to the Apple's machine language monitor through $FF69 (-151). Another **CTRL-Y** returns you to the CP through the warmstart location, $806. This feature is of interest only to the advanced programmer.

CALL addr—Calls the machine language routine at 'addr,' a given signed decimal or hex address. Again, this is of interest to the advanced programmer.

Input/Output Commands

CP Command	Function
L(OAD) filename	Load 'B' type file into edit buffer
S(AVE) filename	Write 'B' type file to diskette
>filename	Write 'T' type file to diskette
(begin, end)>filename	Extract line range and write to 'T' type file
>>filename	Append to 'T' type file
(begin,end)>>filename	Append range to 'T' type file

<filename	Load 'T' type file
(begin,end)<filename	Load range from 'T' type file
<<filename	Append to memory from 'T' type file
(begin,end)<<filename	Append line range to memory from 'T' type file

L (or **LOAD**) **filename**—Loads a 'B' type file into the text buffer. All DOS parameters are allowed.

S (or **SAVE**) **filename**— Saves, or writes, the contents of the edit buffer as a 'B' type file. Again, all DOS parameters are allowed.

'B' type files are DOS 'binary' type files and are recommended for fast loading and saving of files created with the PIE Text Editor.

The processing of 'T' (DOS 'text') type files is allowed from the Command Processor. This file type is slower to read and write but allows much more flexibility than 'B' type files manipulated with Load and Save commands.

The primary commands for 'T' type files are '<' for input (reading) and '>' for output (writing).

It is sometimes desirable to manipulate only part of a file.This option is available when working with 'T' type files. To specify partial file or buffer processing, precede the command with a parenthetical expression as follows, where 'begin' means the lowest line number and 'end' means the highest line number.

> **filename**—Writes a 'T' type file from the text buffer to diskette. If "filename" already exists, its previous contents are replaced.

(begin, end) > **filename**—Extracts a line range from the buffer and saves it as a 'T' type file.

>> **filename**—Appends the contents of the text buffer to the end of an existing 'T' type file.

(begin, end) >> **filename**—Appends a line range from a file to the end of a 'T' type file.

< **filename**—Reads a 'T' type file into the text buffer, clearing the previous contents of the file.

(begin, end) < **filename**—Reads a line range from a 'T' type into the text buffer.

<< **filename**—Appends a 'T' type file to the end of text in the text buffer, which remains intact.

(begin, end) << **filename**—Appends a line range from a 'T' type file to the end of text in the text buffer.

It may be desirable to prefix line numbers to every line or to a range of lines in a file before writing those lines as a 'T' type file:

> **filename**—Saves a 'T' type file to diskette and prefixes each line with its corresponding line number (line numbering is identical to that in the Text Editor). For instance, the sixth line of unformatted text in a file would be saved on diskette with the number 6 preceding it, using this command.

(begin, end) > **filename**—Extracts a line range from a file and saves it to a 'T' type file, each line prefixed with its corresponding line number.

Commands that read or write a 'T' type file may be used optionally with slot numbers in place of 'filename.' For instance:

> # **slotnumber**—Sends all the characters in the text buffer to the numbered slot. If no card is installed in the slot, or if the card has no ROM to handle such a function, the system will 'hang.' Press RESET to redisplay the CP prompt.

< # **slotnumber**—Brings characters from the specified slot directly into the edit buffer, clearing its previous contents. This can be a little confusing visually because you cannot see the characters coming into the buffer, and the only signal of a completed sequence is the reappearance of the flashing cursor. If it does not appear as expected, press **RESET** to redisplay the CP prompt.

(begin, end) < # **slotnumber**—Inputs a line range from the specified port to the text buffer.

Any ASCII character can be designated as the end of file marker for input or output using the '<' or '>' commands. The selected ASCII character is represented using either **@xx** (decimal) or **@$xx** (hexadecimal) as a marker in the following examples, where 'xx' is a valid decimal or hexadecimal number:

(ASCII marker)> # **slotnumber**—Sends the entire contents of the text buffer to a specified port until the specified ASCII character is encountered, at which time transmission ends and the CP prompt reappears.

 # (ASCII marker) > # slotnumber—Sends the contents of the text buffer to a specified port, each line preceded by its corresponding line number, until the end of file character is encountered.

<< (ASCII marker) # slotnumber—Appends the contents of a file brought in through a specified port to the end of the text buffer until the ASCII marker is encountered.

As an example, suppose that you have placed the character CTRL-A halfway through your file. The ASCII Reference Chart (see p. 94) tells you that the hexadecimal notation for CTRL-A is $81 (129 decimal). You might save the first half of your file to a filename or slot that you indicate, using one of the forms below:

@ $81 > filename
or
@ 129 > filename

@ 129 > #1 (going to a printer in slot 1)

@ 129 > #2 (going to a modem in slot 2)

For the programmer, the ability to write a special character by character output 'driver' is provided with the following syntax:

> $addr

This simply means that if the user has a routine in place at hexadecimal address $addr, each character in the input buffer will be passed to the routine as though it were a slot. The routine should return with an 6502 RTS instruction.

Auxiliary Commands

The auxiliary commands are called from the Command Processor by prefixing a '%' (percentage mark) to the commands shown below. The PIE Writer diskette must be in the defined program slot, drive, and volume (the defaults are 6, 1, and 0) to use the auxiliary commands. (This is because the routines that execute these commands must be loaded from the diskette.)

The first auxiliary command issued causes the disk drive's IN USE lamp to light while the CP.AUX file is loaded from the system diskette. Thereafter the full complement of commands below are available from the command processor until EDIT or FORMAT is invoked.

This means that each time you return to the CP and issue a %-prefaced command, the CP.AUX file is loaded. This file adds a great number of useful facilities to the PIE Writer system.

All the commands must be followed by a **RETURN**.

CP Command	Function
%Q	Quit
%C	Catalog and space on diskette
%H	Help
%FS	File status
%WC	Word count
%LC	Line count
%B	Backup current file
%M	Return to main menu
%TS filename	Save tab settings
%TL filename	Load tab settings
%CI	Caps inverse
%CN	Caps normal
%CB n	Break lines to length 'n'
%CS n	Slice lines to length 'n'
%CJ	Join broken lines
%LF (-)	Set/reset line feed after RETURN on output
%D filename	Delete file
%L filename	~~Load~~ Lock file
%U filename	Unlock file
%V filename	Verify file
%R f1,f2	Rename file
%E filename	Exec file

CP Command (cont.)	Function (cont.)
%ST	Start double time
%SP	Stop double time
%RS	Resume double time

%Q—*Quit*. Exits the program to the "]" BASIC level. To "softstart" without losing the contents of the text buffer, type **CALL 2054** and press **RETURN**. The softstart will not work after loading a BASIC program.

%C—*Catalog and space on diskette.* Displays a standard catalog plus the number of unused sectors remaining on the diskette.

%H—*Help.* Loads the file PIE HELP into memory and enables the Editor command **ESC H RETURN**. Reduces available Editor memory by 1024 characters. This memory can be recovered, and the Help command disabled, by typing **%H-**. The Help screen will remain loaded when switching between PIE and FORMAT.

The standard screen display when entering CP from either PIE or FORMAT shows the last valid title, the file length in characters, and the remaining memory in the text buffer. The next three auxiliary commands provide additional information concerning a file's status:

%FS—*File status.* Provides the same text buffer file status report you receive when transferring to the CP from either PIE or FORMAT, plus the beginning, ending and buffer end addresses in decimal and hexadecimal, the number of lines, and the number of words in the text buffer file.

For example, the %FS command returns the following report on an empty text buffer:

```
FILE: FILENAME

LENGTH:          0
MEMORY LEFT:     22015

TEXT START:      16384     ($4000)
TEXT END:        16384     ($4000)
BUFFER END:      39200     ($95FF)

# LINES          0
# WORDS          0
```

These last two lines of information can be requested separately:

%WC—*Word count.* Reports on the count of words (character groups separated by the ' ' character) in the text buffer.

%LC—*Line count.* Reports on the count of lines in the text buffer (word groups followed by a RETURN character).

%B—*Backup current file.* Saves the text buffer to diskette with the current remembered filename, after first renaming the diskette file of the same name to 'filename'.BAK. A previous backup file of the same name will be deleted. (The %B command actually EXEC's the system text file BACK-UP.)

%M—*Return to main menu.* Causes the program PIE WRITER: WORD PROCESSING to be run, redisplaying the System Menu. It is used to access PIE Configure or FORMAT Configure.

%TS filename—*Save tab settings.* Allows tab settings to be saved to a diskette file under any name. If tab settings are an important part of your file you might do the following:

```
COMMAND?: S FILENAME <return>
COMMAND?: %TS ?.TABS <return>
```

The above sequence saves the contents of the text buffer to the diskette under 'filename' and then subsequently saves the tab settings to 'filename.TABS'.

%TL filename—*Load tab settings.* Allows tab settings to be loaded from a diskette file. The new tab settings replace any default tab settings. A possible sequence is:

```
COMMAND?: L FILENAME <return>
COMMAND?: %TL ?.TAB <return>
```

This loads the desired file and then loads the tab settings saved previously under 'filename'.TABS.

%CI—*Caps inverse.* Converts all the characters in the text buffer so that capital letters are displayed in inverse capitals and lowercase is displayed as normal capital letters. This is the standard PIE Writer shipment configuration for an Apple II without a lowercase adapter.

%CN—*Caps normal.* Converts all the characters in the text buffer for normal upper and lowercase screen display.

%CB n—*Break lines to length 'n.'* The command means literally "convert and break." It is used when a file of unknown line length has been loaded from diskette or when the line length is known to be longer than the text buffer length. This can be true in the following circumstances:

- *The file is a BASIC listing in text ('T') file format, with lines frequently exceeding 64 characters (128 for 80- column versions).*
- *The file was created using an 80-column board version of PIE Writer and will be edited using a standard 40-column version.*
- *The file is a data file created by a BASIC program and exceeds 64 characters.*

The effect of the %CB command is to shorten any line longer than PIE's screen width, splitting the line at a word break and placing the balance on a new line. Whenever a break is forced, a CTRL-Z is inserted in the buffer as a marker. See %CJ below for using this marker.

The %CB may be followed by an optional numeric value. If you wish to shorten a file to a specific maximum line length (this example specifies 50 columns), do the following:

```
COMMAND?:%CB 50 <return>
```

%CS n—*Slice lines to length 'n.'* There are times when lines need to be broken at a given column, but do not need to be rejoined. This break is permanent.The %CJ command will not restore the previous appearance of a file broken with %CS.

%CJ—*Join broken lines.* The command joins any two lines divided by a CTRL-Z, removing the CTRL-Z while in operation. It restores the possibility of the text buffer retaining line lengths greater than the text buffer width.

%LF (-)—*Set/Reset line feed after RETURN on Output.* This command instructs the CP to send a line feed after each carriage return to a port when using the text file output command (>). This is often the quickest way to review text in printed, but not formatted, form. As shipped, the %LF default is set to '-,' meaning that no line feed will be sent out after each RETURN unless you enter this command.

All DOS commands, such as DELETE, RENAME, LOCK, and UNLOCK are immediately accessible from the CP. Shorthand versions of several DOS commands are available as auxiliary commands.

All of the commands below are executed through the CP.AUX file. When the first command is entered the disk drive whirrs and the CP.AUX file is loaded. Once the file is loaded, all commands execute immediately.

All of the disk commands allow the use of the DOS Slot, Drive, and Volume specifications. The ? token for the remembered filename is also allowed using these commands (the ? cannot be used with the normal DOS commands).

%D filename—*Delete file.* To delete any file, use the following form:

```
COMMAND?:%D filename
```

You may also use the ? command to simplify the same function, assuming it is the current file that you wish to delete:

```
COMMAND?:%D ?
```

%L filename—*Lock file.* This command prevents you from destroying essential information by denying write access to the named file.

%U filename—*Unlock file.*

%V filename—*Verify file.* This command verifies the internal consistency of a file stored on diskette. If an error is reported, save the file again to a different diskette.

%R f1, f2—*Rename file.* Rename a file by inserting your old filename for 'f1' and your new filename for 'f2' (for example: %R ?, ?.BAK).

%E filename—*Exec file.* This is the appropriate way to initiate 'shell' commands. See p. 149 for a discussion of how to use the shell file command feature.

If you own the Doubletime Printer spooling software, the following commands allow you to control background printing directly from PIE Writer. You should install the Doubletime Printer prior to using PIE Writer.

Hayden Software does not support the Doubletime Printer, but PIE Writer includes the following helpful commands.

PIE Configure must be informed that Doubletime has been installed. The file DOUBLETIME PRINTER, which should be transferred onto the PIE Writer system diskette, is loaded the first time the **%ST** command is entered.

%ST—*Start Doubletime.* This command performs the START operation of Doubletime, beginning the printing of any valid files ending with a "D."

%SP—*Stop Doubletime.* This performs the STOP operation of Doubletime, halting further output without altering spooling "pointers."

%RS—*Resume Doubletime.* This RESTARTs Doubletime for continuation after a STOP. The %SP %RS sequence is used to stop printing temporarily and then continue on as before.

FORMAT TEXT PROCESSOR

FORMAT dot commands allow you to control the layout of your document. The FORMAT Text Processor acts upon the instructions relayed through the dot commands and outputs formatted files to a printer, a diskette, or the screen.

Dot commands may be typed in either upper- or lowercase, but each command must be typed on a new line with the dot in column 1.

The letter 'n' that follows many of the dot commands is a variable. You may replace the 'n' with a new value, or you may leave it off. By prefixing 'n' with a '+' or '-,' the current 'n' value is increased or decreased accordingly by the value that follows.

Some commands cause the line that follows them to "break," meaning that the FORMAT Text Processor places text following those commands on a new line, instead of continuing the paragraph above.The commands that cause text breaks are noted with an asterisk*.

The "Default" column indicates the value PIE Writer assumes if a command that takes a variable is typed without entering a value.

The "Initial" column indicates the 'n' value that the FORMAT Text Processor assumes to be a valid formatting parameter value if a dot command is never given to change that value. For instance, if you never specify page length with the **.pl** command, FORMAT assumes that you want a 66-line page.

Both the default and initial values for many of the dot commands may be redefined by selecting the FORMAT Configure option from the System Menu.

The following are the dot commands that FORMAT recognizes and executes.

Paragraph

The **.pp** and **.np** commands signal the beginning of paragraphs and activate other layout and indentation parameters within FORMAT. The **.ps**, **.pn**, and **.pi** commands override these parameters and go into effect whenever subsequent **.pp** or **.np** commands are given.

Command	Default	Initial	Function
.pp n *	n = +5		Begin paragraph
.ps n	n = 1	1	Modify paragraph spacing
.pn n	n = 2	2	Lines needed to begin paragraph
.pi n	n = +5	+5	Paragraph indent
.np n *	n = 2		Begin no-fill paragraph

A **.pp** command begins a paragraph and activates the following four built-in commands with specific variables:

.sp—There is one line of space between consecutive paragraphs.

.ti +5—The first line of each paragraph is temporarily indented 5 additional spaces.

.ne 2—There must be enough room for at least 2 lines of a paragraph that begins at the bottom of a page, or else the entire paragraph begins on the next output page.

.fi—All paragraphs are output in the fill mode.
Enter a **.pp** command immediately preceding each paragraph you want to be output in typical fashion. Following the **.pp** command with a variable indents the first line of each paragraph 'n' spaces instead of the default value.

The .np command assigns the 'n' variable to the built-in **.ne** command. The no-fill paragraph command also activates three built-in commands:

.sp—There is still one line of space between consecutive paragraphs.

.ne n—The number of lines needed to the bottom of the page before beginning a paragraph on a new page equals 'n.'

.nf—All text is entered in the no-fill mode until a .pp or .fi is encountered.

No-fill paragraphs are the preferred setting when preparing the output of any material that you want to print exactly as it appears on the PIE screen. The default value for 'n' is 2, meaning that two lines are needed at the bottom of a page or else the no-fill paragraph will not be started until the top of the next page.

The **.ps**, **.pn**, and **.pi** commands override parameters used by **.pp** or **.np**, and they are activated by every **.pp** or **.np** command that follows them.

.ps—*Modify paragraph spacing*. This command overrides the .sp default and initial value to increase, decrease, or eliminate space between paragraphs.

.pn—*Lines needed to begin paragraph*. The .pn overrides the .ne default and initial value to increase or decrease the number of lines needed to begin a new paragraph at the bottom of a page.

.pi—*Paragraph indent*.This command overrides the .ti default and initial value to alter each paragraph's first line indent.
Once entered, these three commands are carried out every time you enter a **.pp** or **.np** command thereafter. (The **.pi** is ignored by the **.np** command.) You can 'permanently' change the initial **.ps**, **.pn**, and **.pi** values by altering values in FORMAT Configure's Paragraphs and Filling option.

Layout

The layout dot commands control text filling and adjusting modes, vertical spacing, and page control.

Command	Default	Initial	Function
.fi *		yes	Fill
.nf *		no	No-fill
.ad *		no	Adjust right
.na *		yes	No adjust right
.br *			Break in filling
.ls n *	n = 1	1	Line spacing
.sp n *	n = 1		Space down 'n' lines
.bl n *	n = 1		Force 'n' blank lines

Command	Default	Initial	Function
.pl n	n = 66	66	Page length
.bp n *	n = +1	1	Begin page
.po n	n = 0	0	Page offset
.ne n	n = 1		Lines needed to bottom

The FORMAT Configure option in the main menu is shipped with the fill mode set, so if neither the **.fi** or **.nf** commands are inserted, the **.fi** will be assumed.

.fi—*Fill*. In this mode, FORMAT gathers as many words as it needs to fill a line of output text. This mode is useful in typical document preparation, allowing you to enter text continuously in PIE following a .fi command, with the assurance that FORMAT will reorganize the text into lines of the length you specify. FORMAT 'fills' up lines by gathering words from the input file, until it encounters either a break command (.br) or any command that causes a break (indicated by the '*'). A blank input line or a line beginning with a space also causes a break.

Fill mode automatically turns a series of spaces into a single space, except in two cases. If you type two or more spaces after a period, colon, question mark, or exclamation mark, or if one of these characters is the last in a line, then it is treated as the end of a sentence, and at least two spaces are output.

There is also a way to insert 'unpaddable' spaces that are not eliminated in the fill mode (see 'Character Translation and Definition' on p. 133).

.nf—*No-fill*. When working on a document in which output lines should be broken exactly the way you enter them in PIE, the no-fill mode is the setting to use. It does no rearranging and ignores the current line length. Precede text that should be formatted as entered, such as outlines or tables, with a .nf command.

.br—*Break in filling*. In fill mode, the .br command tells FORMAT to start subsequent text on a new output line. In no-fill mode, .br has no effect.

Justification commands allow you to select either right justification or ragged right text. The default is ragged right.

.ad—*Adjust right*. In fill mode, this command begins the right justification of text that immediately follows it, according to your specified line length requirement. If your printer is capable of incremental spacing, it does that as well, provided you have specified the appropriate printer type in FORMAT Configure.

.na—*No adjust right*. This command cancels right-margin justification and produces a ragged right margin.

Vertical line spacing is controlled with the **.ls**, **.sp**, and **.bl** commands:

.ls—*Line spacing*. This command sets the spacing between lines, putting one less than 'n' spaces between each line (for instance, .ls 1 gives single spacing, .ls 2 gives double spacing, and so on). The default is for single-spaced documents.

.sp—*Space down 'n' lines*. Insert as many blank lines in your document as you wish with this command. If there are less than 'n' lines left on the page, only the number of line spaces that fit will be output.

.bl—*Force 'n' blank lines*. You can insert blank lines in your document with the .bl command also. The only difference between the .bl and .sp commands is that the .bl forces 'n' blank lines even if FORMAT must break to a new page to insert all the blank lines.

Additional page specifications are controlled with the **.pl**, **.bp**, **.po**, and **.ne** commands. The **.pl** and **.po** parameters can be redefined by entering different values using FORMAT Configure.

.pl—*Page length*. This command tells format your page size, up to a maximum length of 250 lines. The default value for 'n' is 66, assuming a standard page depth of 11 inches and a standard 6 lines per inch.

.bp—*Begin page*. The .bp command tells FORMAT to begin the text that immediately follows it on a new line at the beginning of a new page.

The initial value for 'n' is 1. Used in conjunction with the head (.he) and foot (.fo) title and '%' page numbering capability, however, you can begin page numbering on a page other than the first. For instance, to insert numbers at the top right-hand corner of a page, beginning with the twelfth page, enter a .bp 12 to begin a new page and set its number to 12, followed by an .he'''% command.

.po—*Page offset*. The .po command allows you to set up the left-hand margin of your documents for your particular printer. The page offset is the area, in characters, at the left side of your page that your printer's print head

must be advanced beyond to begin the left-hand margin. (Alternatively, some printers allow manual adjustment of the print head's 'home' position.) The default value for 'n' is 0.

.ne—*Lines needed to bottom*. If there is not enough room to fit at least 'n' lines of text at the bottom of a page, the text that follows is forced to begin on a new page. The default value for 'n' is 1.

Since the .ne command does not cause a break in fill mode, you may want to precede it with a .br command. This places all previous text at the bottom of a page before the .ne command checks to see if enough lines are available to output the text that follows it.

Indentation

Elements that may require structured left and right indents, such as extracts, outlines, or lists, can be formatted with the **.ll**, **.in**, and **.ti** commands. The initial setting for the **.ll** command can be altered by reconfiguring a selection in FORMAT Configure's Page Format option.

Command	Default	Initial	Function
.ll n	n = 65	65	Line length
.in n *	n = 0	0	Indent
.ti n *	n = 0	0	Temporary indent for next line

Plus and minus values can increase or decrease a current value for any FORMAT dot command that takes an 'n' variable. An indenting command without a plus or minus prefix to the 'n' value sets the indent to an absolute column. For instance, an **.in 5** command sets the indent 5 spaces to the right of the left margin; an **.in +5** command sets the indent 5 spaces to the right of the current indent.

.ll—*Line length*. The .ll command allows you to determine the length of each output line of text, up to 132 characters long. The .ll command can be used to indent text from the right-hand side of a page. For instance, to indent an extract of quoted material 5 characters from the left and right margin, precede the quoted material with .in +5 to indent the left side 5 spaces and .ll -5 to indent the right side 5 spaces.

.in—*Indent*. This command indents all text that follows it 'n' character spaces from the left margin. Plus or minus values are valid 'n' variables, so outlines and lists, elements that can have constantly changing indents, are best formatted using the .in command.

.ti—*Temporary indent for next line.* You can indent a single line of text, then return all subsequent lines to their previous indent. For instance, you can outdent the number of each item in a list and still left-justify the items themselves (see p. 134 for more on outdents).A plus 'n' value allows you to indent a line of text 'n' additional character spaces to the right, then return the alignment of all subsequent lines to the previous indent.

Running Titles and Margins

The **.he** and **.fo** commands allow you to create running titles that appear at the top or bottom of every page of your document; the **.tl** command lets you place a single title anywhere in your document. These commands are most useful when specifying running heads or the consecutive numbering of pages.

Command	Default	Initial	Function
.he t	t = ''''	''''	Head title
.fo t	t = ''''	''''	Foot title
.tl t	t = ''''	''''	Flush left, center, flush right

.he t—*Head title.* Text that follows the .he command on the same line is output at the top of every page of your document. You can change the head title with the .he command anywhere on the page preceding the one on which you want it to appear. The top margin defaults (see the .m1 and .m2 commands below) print the head title 1 line from the top of the page and 2 lines above the text. The head title may be moved by adjusting the margins.

.fo t—*Foot title.* Print a title at the foot of every page of your document with the .fo command. The title that you want printed must follow the .fo command on the same line. The foot title can be altered anywhere before you want it to appear. The bottom margin defaults (see p. 125) print the foot title 1 line below the text and 2 lines from the bottom of every page. You can adjust these margins to move the foot title.

.tl t—*Flush left, center, flush right.* The title following the .tl command on the same line is printed once only, exactly where it is entered in the text. The .tl command could be used to right-justify the date in a letter or to center a subhead.

Unlike the .ce command (see p. 126), the .tl does not cause a break. A .tl in the middle of a fill-mode paragraph results in filled lines both above and below the centered material.

If no title 't' is specified, the .tl command outputs a blank line.

The pattern for entering all three title commands is the same: up to three "parts" of a running title, separated by any limiting character (any character that does not appear in any of the parts), can be entered. The position in which they are output is Part 1 begins at the left margin; Part 2 is centered; and Part 3 is flush right.

To set all three parts in a running title, simply separate them with a valid delimiter (such as an apostrophe, in the example **.fo 'Part 1'Part 2'Part 3'**). To omit any part of the running title, simple omit that part, leaving the delimiters intact (for instance, **.he '''Part 3'** prints only "Part 3" flush right at the top of your page). You need not insert a delimiter to the right of the last section you want printed, though (for instance, **.tl 'Part 1** is the correct command for inserting "Part 1" flush left on the next line on the page).

You can instruct FORMAT to page your document for you automatically by using the **.he** or **.fo** command with the percent sign. For instance, to have the page number appear in the top right-hand corner of every page of your document, beginning with page 1, you would enter the **.he '''%** command before entering any page 1 text. To begin paging your document anywhere after the first page, you must set the page number with a **.bp** command, which specifies that paging should begin on the next page, page 'n.' The percent sign in the title command pages your document beginning with page 'n' and increments its value by 1 to page continuously to the end of the document.

The maximum line length for **.he** and **.fo** titles is the text line length at the time the **.he** or **.fo** title is specified.

Head and foot titles are unaffected by indents or temporary indents. However, either of the first two parts of a head or foot title can be shifted right by inserting spaces between the delimiter and the beginning of the part.

Four margin commands allow you to change the vertical position of your head and foot titles. The initial values can be changed "permanently" by altering selections in FORMAT Configure's Top and Bottom Margins option.

Command	Default	Initial	Function
.m1 n	n = 2	2	First top margin (to head)
.m2 n	n = 2	2	Second top margin
.m3 n	n = 1	1	First bottom margin (to foot)
.m4 n	n = 3	3	Second bottom margin

.m1—*First top margin (to head title).* The 'n' value that you enter with the .m1 command determines the number of lines from the top of the page down to, and including, the head title. If you insert a head title without a .m1 command, the head title will be output 1 line from the top of your page. Since the .m1 value includes space for the head title, setting .m1 to 0 "turns off" the header.

.m2—*Second top margin.* This 'n' value specifies the number of lines between the head title and the first line of text, excluding the first line of text.

.m3—*First bottom margin (to foot title).* This margin dictates the number of lines between the last line in the main body of your text and the foot title, excluding the foot title.

.m4—*Second bottom margin.* The fourth margin tells FORMAT how many lines of space there should be between the bottom of the overall page, up to, and including, the foot title. Just as .m1 0 turns the head title off, .m40 turns the foot title off.

If none of the four margin commands are entered anywhere in a file and no head or foot titles are specified, FORMAT prints your document with 4 lines of space at the top and bottom of every page.

Finishing Touches

PIE Writer is capable of centering, boldfacing, underlining, and capitalizing text anywhere on your page. This is helpful in placing emphasis on text within a line or in conveying a document's organization with heads and subheads.

Use FORMAT Configure's Printer Interface option to set up the underlining and boldfacing techniques that fit your printer.

Command	Default	Initial	Function
.ce n *	n = 1	0	Center
.bf n	n = 1	0	Boldface
.ul n	n = 1	0	Underline
.cp n	n = 1	0	Capitalize

.ce—*Center.* This command centers the following 'n' lines of unformatted text. Each line is centered individually, whether the command is entered while in the fill or no-fill mode.

.bf—*Boldface.* The .bf command restrikes text or changes into a boldface type font for text (depending on your printer) on the following 'n' lines of text.

This command allows you to boldface even a single word in the middle of a fill-mode paragraph.

.ul—*Underline.* Precede the 'n' lines of text you want underlined with the .ul command.

.cp—*Capitalize.* You can capitalize the following 'n' lines of text by immediately preceding them with the .cp command.

File Switching and Form Letters

The following set of FORMAT dot commands facilitate file switching, particularly useful when an entire single document, such as a long report, would be too large to fit in the text buffer, and for easy form letter production.

Command	Default	Initial	Function
.nx filename	prompts		Chain to next file
.op filename	prompts		Open data (secondary) file
.rb -	space		Read block ('-' optional)
.sb n	n = 1		Skip blocks
.ng			Skip blocks to end of group

The text buffer has enough room for approximately 5 1/2 pages of output text (assuming 65 characters per line of text, 60 lines of text per output page). Any document that exceeds that length does not fit in the text buffer. The **.nx** command allows you to string together files for continuous document output.

Each 'filename' specified in a **.nx** or **.op** command is converted to uppercase.

.nx filename—*Chain to next file.* Insert the .nx command at the end of any file to chain to another file (named "filename"). When the FORMAT Text Processor has finished formatting the first file, it loads the "filename" automatically (or if a filename is not specified it sends a prompt to the screen asking for the file's name). The second file 'filename' will be output immediately following the first file, as if the text from both files were actually in only one file. Many files (an entire book manuscript, for example) can be chained together in this way.

If 'filename' is not stored on your diskette, FORMAT outputs the first file until it reaches the .nx command, then interrupts the output process and displays FILE NOT FOUND.

When FORMAT encounters the first .nx command, the current contents of the text buffer are saved in a file named TEMP*** before writing over the text in the buffer with the second file. The original contents are restored in the buffer when the entire document has been output.

All commands and text that follow the .nx command in the first file are ignored.

The **.op** command allows a primary file to access a secondary data file (for instance, containing a mailing list) and to merge "blocks" from the secondary file into the primary document with **.rb**, **.sb**, and **.ng** commands. Using these four commands in your primary file allows you to produce accurate form letters easily.

.op filename—*Open data (secondary) file.* Normally the data file contains names and addresses, phone numbers, and any other information that you would want to manipulate for insertion in a form letter. The .op file can contain formatting commands as well, except for any of the commands that access it (.op, .rb, .sb, and .ng). An .nx command can be inserted at the end of a secondary file to chain together a long mailing list, for instance.

While the .op command can be inserted anywhere prior to the block manipulation commands in your primary file (the file containing the stable, main body of your form letter), it is best to place it at the beginning.

If there is no file named 'filename' existing on diskette, the FILE NOT FOUND message is displayed after output is interrupted when FORMAT encounters the invalid .op command.

Two secondary data files cannot be opened simultaneously from the same primary file. The second .op command encountered in a form letter is ignored.

Entering the **.op** command in the primary file does not access a secondary data file, it only opens it. To read any information from a secondary file, the **.rb** command must appear in the primary file anywhere after the **.op** command. The **.rb** command should appear, however, at the exact point in the primary file where data from the secondary file, such as an address, must be inserted.

.rb -—*Read block ('-' optional).* This command instructs FORMAT to read a block of data from the file opened with the .op filename command. Data blocks are defined by the block terminating character, which follows the last line of information in a block. The initial block terminating character is <, but this can be replaced "permanently" with any character you choose by changing the fifth value in FORMAT Configure's Lowercase and Special Character option, or temporarily with the .bc command (see p. 133).

For instance, to insert an address from a data file named ADDRESS as the inside heading in the primary file, the primary file might look like the one below:

```
.OP ADDRESS
.NF
TODAY'S DATE
.SP
.RB
.SP
.FI
DEAR
.RB -
,
```

The data file ADDRESS would look like this:

```
MS. MARKETING DIRECTOR
HAYDEN BOOK COMPANY, INC.
50 ESSEX STREET
ROCHELLE PARK, NJ 07662<
MS. DIRECTOR<
```

The execution of the above commands and text would insert the address where the first **.rb** command is given, providing an inside heading to a typical form letter, and the name where the second **.rb** command is given. The '-' following the second **.rb** tells FORMAT that in fill mode the characters on the next line should be appended to the character preceding the '<'. In this case, the comma immediately follows "MS. DIRECTOR."

Obviously, data files are much more useful if they contain several blocks of information rather than just one. For instance, a data file might reasonably contain not only a company's name and address, but its phone number, telex number, and so on. Defining each of these pieces of information as a block, with a block terminating character, would allow you to use several **.rb** commands in your primary file and insert blocks throughout a form letter.

Data blocks in a secondary file can be skipped with the **.sb** command. This command is most useful when compiling a form letter that requires only certain blocks of information in a multipurpose data file.

.sb n—*Skip blocks.* Any number of blocks, specified by 'n,' in a data file can be skipped during the output of a primary file with the .sb command. Any commands (including the .nx command) in the skipped blocks will not be executed.

One typical application for the .sb command would be in reading blocks for a form letter from a data file also used to print mailing labels. The primary file might look like this:

```
.OP ADDRESS
.NF
TODAY'S DATE
.SP
.RB
.SB
.SP
.FI
DEAR
.RB -
,
```

The data file named ADDRESS might appear as shown below:

```
MS. MARKETING DIRECTOR<
HAYDEN BOOK COMPANY, INC.
50 ESSEX STREET
ROCHELLE PARK, NJ 07662<
.SP
ATTN: MARKETING DIRECTOR<
MS. DIRECTOR<
```

Executing these two files would insert the marketing director's name and the company name and address in the appropriate position in the primary file form letter, ignore the information applicable only to printing mailing labels, then pick up the name and comma for the salutation.

Outputting the primary file via FORMAT produces as many form letters as there are "groups" of data blocks, then ends transmission. In other words, a data file with 10 groups of 5 data blocks provides 50 inserts, enough material for 10 letters. Blocks in the data file are read or skipped every time an **.rb** or **.sb** is encountered until the data file is exhausted.

Each company name in a data file, for instance, along with an address block, a mailing direction block, a phone number block, and a telex number block, constitutes one "group" of blocks. In the event that you mistakenly form a data file with 9 groups of 5 data blocks and 1 group of 6 blocks, PIE Writer provides a command that synchronizes each group of blocks.

.ng—*Skip blocks to end of group.* Just as the < character signals the end of a data block in a data file, << signals the end of a group of data blocks. Insert the .ng command on the last line of your primary file to make FORMAT search for the << signal following the last line of a group of data blocks. Then insert the << signal at the end of each group of data blocks.

If FORMAT has not encountered the << signal by the time the primary file output is completed, FORMAT ignores any data blocks that have not been read (.rb) or skipped (.sb) by the primary file and advances to the next group for doing the next form letter. This provides a synchronizing feature to overcome occasional typing errors (when one group is longer or shorter than the others) and limiting erroneous output to only a single label, letter, or document.

See "Mail Merge" on p. 139 for a more complete discussion of PIE Writer's form letter capabilities.

Interactive Input and Control

A second method for inserting material in primary files is via the terminal communication commands, which are also used to halt output temporarily and to send messages to the screen.

Command	Default	Initial	Function
.gl str	str = '?'		Inserts line entered at keyboard
.st str	str = ''		Stop (any key continues)
.tm str	str = ''		Output message to terminal
.sn		no	Stop next page

.gl—*Inserts line entered at keyboard.* The .gl command is used to insert text or any command (except .op, .rb, .sb, and is being output through FORMAT.

When FORMAT encounters a .gl command, output is interrupted, and the string ("str") that follows the command is output to the screen (if no string accompanies the .gl, the screen displays a question mark, the default, to remind you of what needs to be input.) Respond to the display with text or a command entry, then press RETURN. That entry is then inserted at the point in the file where the .gl command appears.

In other words, the .gl command gives you the power to create a prompt asking for a specific response. Your response to the prompt is inserted in the file, which is then formatted and output.

For instance, you can use the .gl to obtain often- changed text from the keyboard. If your primary file contains this text:

```
.FI
DEAR
.GL ENTER NAME AND COMMA
.SP
BODY OF LETTER HERE.
```

The .gl then will insert the name and comma you enter after "Dear." (If the no-fill mode were set, your entry would appear at the beginning of the next line.)

Lines also can be joined together when a] followed by a **RETURN** is inserted at the end of a line (see how to set and use the escape character on p. 134). This can be particularly useful when obtaining command variables from the keyboard. If your primary or data file contains the command sequence below:

```
.LL]
.GL LINE LENGTH?
```

A response of 2 to the prompt will have the same effect as if a .ls 2 command were given in the original file.

.st—*Stop (any key continues).* This command interrupts the formatted output of a file and displays whatever string ("str") follows the .st on the same line. The .st does not allow interactive prompt-and- response, as does the .gl command, but only interrupts output until you press any key.

One useful application for the .st command would be to provide a pause in output for making a physical adjustment to your printer. The .st command does not cause a break, so in fill mode, the interruption of output occurs only at the end of a filled line. Used in conjunction with the .br command, though, the .st could be used as shown below:

```
.BR
.ST CHANGE TO RED INK
```

Additional information about this use of the .st command appears in the "Terminal Communication" section on p. 146. A more precise means of performing the same function is provided with the .tm command and the]+ sequence.

.tm—*Output message to terminal.* The .tm command performs the same function as the .st command, except that it does not interrupt the formatted output of a file. The string ("str") accompanying the .tm is displayed whether your file is output to the printer, diskette, or screen.

Used with the]+ signal, the .tm command can interrupt transmission at a precise point in a file and display the accompanying message, allowing you to make physical printer adjustments that produce accurate changes, for instance, within a line of text. See the "Terminal Communication" section on p. 146 for discussion of a situation in which precise output interruption is desirable; see "Character Translation and Definition" below for additional information on escape characters.

.sn—*Stop next page.* The .sn command performs a temporary stop at the end of the output page it appears on, whether it is inserted in a primary or data file. A blinking cursor appears after transmission is interrupted. Press any key to resume output.

Character Translation and Definition

The character translation commands, .* and .li, let you enter remarks and dot commands in the PIE Text Editor that are not formatted when a file is output. The character definition commands, .ec and .bc, let you enter special characters for FORMAT, printer, or data file merging instructions.

Command	Default	Initial	Function
.*			Comment line (not output)
.ec c	off	']'	Escape character
.bc c	= RETURN	'<'	Block terminating character
.li n	n = 1	0	Literal lines follow

.*—*Comment line.* Preceding any text message with the .* command allows you to enter comments in the PIE Text Editor that are not output during formatting. This can be particularly useful for leaving yourself editing notes, while ensuring that the notes will not be seen on printouts.

For example, you might wish to insert notes in an invoice file, such as:

```
.* INSERT DATE LINE 10; CURSOR # LINE 14
```

or

```
.* TABBING BASED ON $000.00
```

The comment command is useful also when you want to insert source references in documents. This allows you to keep a record of all your sources on diskette, but to leave them out of the formatted document itself. A typical entry might be:

```
.* REF. PERSONAL COMPUTING, JANUARY 1983,
.* "WORD PROCESSING SOFTWARE," VOL. VII, NO. 1
.* PP. 25-30.
```

Print any files containing .* commands via the Command Processor for a hard copy of your program listing with comments.

.ec—*Escape character.* The escape character is used to send instructions to FORMAT or to your printer regarding specific activity regarding the character or line that follows it.

You can change the escape character default from] (SHIFT-M) to any other character you want simply by preceding that other character with an .ec command.

Normally, the escape character disappears when formatted.To print it out, enter it twice (for example,]]). If you type .ec without specifying a character, all escape sequences are disabled.

The escape sequences are shown below:

'] '—*An unpaddable (mandatory) space, not to be adjusted.* Guarantees one, and only one, space. (Two in a row guarantees two spaces.) Words separated by unpaddable spaces are never split across, but are kept on the same line together.

The escape character (]) is not printed.

']>'—*A paragraph tab.* Tabs from temporarily "outdented" paragraph slugs or "side heads" forward to the current indent position by inserting unpaddable spaces. The left alignment of the text to which the side head refers is maintained. For example, this input:

```
.in +15
.pi -15
.pp
a list item]>Paragraph starts here,
after the paragraph tab.  When
formatted, it will be within the
indent specified; the list item
will be 'outdented' at the
negative paragraph indent.
```

Creates this when formatted:

```
a list item          Paragraph starts here,
                     after the paragraph tab.
                     When formatted, it will
                     be within the indent
                     specified; the list item
                     will be 'outdented' at
                     the negative paragraph
                     indent.
```

']RETURN'—*Join next input line.* Allows lines longer than text buffer width. The first character on the next line (and the characters that follow it) immediately follow the character that precedes the]RETURN. No spaces are inserted between the two characters, so if they are separate words, insert a space before typing the]RETURN.

With this feature, you can join several lines in a row by appending a]RETURN to each line. This allows you to create very wide tables in the no-fill mode. Connected lines must not total more than 132 characters. Any characters that exceed this maximum are cut off.

']"chars"'—*Send all chars with 0 width.* Typically used for sending printing characters as part of a printer command string without affecting justification. (For example, to get double strike emphasized using an Epson printer, enter]" <esc> E <esc>G".)

']+'—*Stop formatting at once.* Waits for any key to be pressed to continue.The]+ sequence can be used to stop the printer and change daisy wheels, thimbles, or fonts before continuing with a new typeface.

.bc—*Block terminating character.* You can change the block terminating character from < to any character you wish with the .bc command. If no variable follows the .bc command then each line of your data file will be treated as a block, and a blank line will be treated as the end of a group.

If you change the block terminating character, the group terminating character becomes two consecutive block terminating characters. For instance, if you change the block terminating character to $ then the group terminating character becomes $$ accordingly.

.li—*Literal lines follow.* The .li command lets you output 'n' lines that begins with a period in a file. Dot commands can be entered and printed literally via FORMAT if preceded by a .li command.

ERRORS AND ERROR MESSAGES

When you command PIE Writer to carry out a confusing or incorrect instruction, a system error occurs. PIE Writer will recover from such errors with a minimum of difficulty. Below are examples of situations in which errors occur, along with a description of error messages that may be displayed and the steps that should be taken to recover.

Reset Error

The **RESET** key on the Apple can be a source of great inconvenience. When pressed, it causes the computer to stop whatever it is doing.

If you have one of the older Apples with the old monitor ROM, pressing **RESET** while in the Text Editor sends you to the system monitor, indicated by the * prompt that appears on the screen to the left of the cursor. To recover your text and continue editing, type **809G**. The screen will be restored exactly as it was when you left off.

For the majority of Apple owners who have the autostart monitor ROM, the **RESET** key has been effectively disabled, so pressing it will not cause any problem.

Recovering Text After Rebooting

There are times when, due perhaps to a hardware malfunction, the only way you can regain control of the computer is by rebooting the diskette.To recover text in memory (assuming that the power has NOT been turned off):

1. *Select option* **5** *(EXIT) from the System Menu.*
2. *When you receive the] prompt, type* **BLOAD PIE**, *then press* **RETURN**.
3. *When you receive the] prompt again, type* **CALL 2054** *to "warm start" PIE with the text buffer intact, then press* **RETURN**.

It is recommended that you save the text in memory on a diskette immediately.

Disk Errors

Disk errors are generated by the Apple DOS 3.3 operating system. These errors are trapped by PIE Writer and then reported to the user in messages, such as FILE NOT FOUND and DISK IO ERROR. (A full description of the nature of these errors is found in the DOS 3.3 reference manual.) Generally, by respecifying the command correctly, you can correct the error.

A DISK IO ERROR, however, may mean that your diskette has been damaged beyond recovery (or it simply may mean that your disk drive is open). This is the main reason why making backup diskettes is so important.

The FILE NOT FOUND error can occur after entering an AUX command, typing **F** in the Command Processor to transfer to the Text Processor, or trying to return to PIE from FORMAT. This error is displayed if you remove your system diskette from the drive specified as the program drive (normally slot 6, drive 1). Simply reinsert the system diskette and repeat the command.

Another commonly encountered disk error is FILE TYPE MISMATCH. This messages appears on the screen if you try to update a file that is already on the diskette using, for example, the **S** (save) command after using the > (write) command (or vice versa) to store it on diskette. This causes an error because those two commands create different type files.

You can always update a file using the same filename—the old file of the same name simply will be overwritten—but the old and new files must be of the same file type. The **S** command and the > command create DOS binary and text type files, respectively. (For the same reason, a file saved using **S** must be loaded using **L**, not >.)

6

Advanced Topics

When used to their full capabilities, the editing, layout, and file manipulation commands can make PIE Writer perform powerful and sophisticated word processing functions. The Mail Merge and Terminal Communication features, introduced in the previous section, make printing any form letter an easily managed task if the primary and data files are set up attentively. Shell File Structures allow you to define a frequently used command sequence and perform the consecutive functions with much less typing effort.

Instructions on how to modify your Apple so that the **SHIFT** key provides access to uppercase character entry, as it does on a standard typewriter, are given.

Information on how to transmit PIE Writer files via modem or from Apple to Apple are also included in the Telecommunication section.

Finally, for the programmer who wants to expand PIE Writer's capabilities even further, the Address Tables section provides PIE Writer entry point and parameter addresses.

MAIL MERGE

Mail Merge allows you to establish data files that can be merged into a sequence of documents. Each document output is identical except where the data in the data file changes.

For instance, if you needed to reach a list of clients to announce a change in policy or price, it would be nice to write a single letter and personalize that letter with data from a file of names, addresses, and other information.

FORMAT performs this task easily as long as the primary and data files are created thoughtfully.

Primary File

Below is a listing of a primary file, with comments, that utilizes Mail Merge's best features:

```
.OP CUSTOMER LIST,S6,D2
.NF
.*
.* THESE ARE COMMENT LINES
.*
.* Center company name and address
.* and capitalize company name
.CE 3
.CP
Hayden Book Company
50 Essex Street
Rochelle Park, N.J. 07662
.SP
.* Read store manager's name
.RB
.SP
.* Read store name and address
.RB
.* Set salutation flush left
.NP
Dear  .* Read in first name of manager
.RB
.PP
Hayden Book Company is pleased to
announce the immediate availability
of
.* Underline and boldface title
.UL
.BF
PIE WRITER: Word Processing System
.PP
This powerful addition to the Hayden
line of software will provide your
business and professional users with
the most flexible editing and word
processing system available.
```

```
.PP
No hardware boards or adapters are
required for operation of PIE Writer,
yet most of the commercially sold
peripherals can be "switched in"
under software control.This
includes upper- and lowercase
adapters and 80-column boards.
.PP
PIE Writer works beautifully with
standard Apple II disk drives and
equally well with Corvus and other
hard disk systems.
.PP
PIE Writer includes the following
professional word processing
features:
.SP 2
.* Indent items in list
.IN 5
.NF
PIE (Editor)
.SP
* line and character INSERT
* line and column GOTO
* global search and replace
* MOVE (line or linerange)
* COPY (up to 21 lines)
* Word Tab (forward and backward)
* Word and Line DELETE
* and many more commands
.SP 2
FORMAT (Printer Output)
.SP
* Mail Merge (letter and label)
* Terminal communication
* Video screen preview
* Underline
* Boldface
* Source file switching (for books
.* Line turns on extra indent
.TI +2
```

```
and other lengthy documents)
.TI -2
* Center one or a number of lines
* Automatic paging
* and more
.IN
.SP
.PP
PIE Writer has the power and
flexibility to provide for the needs
of the beginner and the expert.As
your skill and understanding grow,
PIE WRITER grows with you.
.PP
All of this is provided at a cost
less than that of comparable word
processors, which
.UL
.BF
require
the Z-80 CPU card and an 80-column
video display board. This is a real
saving to your customer. If the user
owns a typical 80-column board, PIE
Writer supports it.
.PP
.* Repeat the manager's name
.RB
we would like to answer all of your
questions. Please call or write for
more information.
.NP
Cordially,
.SP 4
Janice Doe
Distribution Manager
.* Protect yourself from printing
.* several incorrect letters
.* with the "next group" synchronizer
.NG
```

You would enter the above document with PIE and save a copy to the diskette under the name PIE WRITER SALES LETTER.

Data File

Now take a look at four sample entries in a file called CUSTOMER LIST. This is a data file of manager's names, outlet names, and addresses of stores that may be interested in selling PIE Writer. Notice that the data file's name was specified in the very first line of the primary file by the **.op** command.

To add more names and addresses, begin after the last <<, place block terminators (<) at the end of each line or block of lines, and place a group terminator (<<) at the end of each data group. Then you may place any comments about the store relative to potential sales or problems in the area between the last block terminator and the group terminator. These comments never appear in the printed document.

```
.* Each < is a block terminator
.*
.* Each .RB in the primary file read
.* everything to the next block
.* terminator and merges it into the file
.*
Timothy Gilbert <
Computer Store of Rome
77 Main Street
Rome, N.Y.99999 <
Timothy, <
Timothy, <
(415) 999-9999 <

Four previous Hayden purchases
Mostly business customers
Performs installations
***Good reputation***
<<
Helen Drysdale <
Computer Store of Paris
777 Prospect Drive
Paris, TX 11111 <
Helen, <
Helen, <
(415) 888-8888 <

No previous purchases
Business and hobby customers
Performs installations
```

```
***Good reputation***
<<
Carl Feinberg <
Computer Store of Moscow
777 Exterior Street
Moscow, PA 55555 <
Carl, <
Carl, <
(415) 777-7777 <

No previous purchases
Business and hobby customers
Performs installations
***Excellent reputation***

Had a good time on 12/14 sales trip
Likes skiing, music
<<
Sandy Phillips <
Byte Shop
100 Decatur Road
New London, CT 33333 <
Sandy, <
Sandy, <

(415) 666-6666 <

Last contact 11/10
No previous purchases
Hobby customers
Does not assist with installations
Fair reputation
<<
```

When the file PIE WRITER SALES LETTER is loaded and formatted, four copies of the letter are printed out, each differing only in the name of the store manager and the store address. All of the comment information from the phone number to the group terminating character is left out of the printed form letter.

At least one copy of each different letter is printed automatically. In response to the FORMAT prompt "OUTPUT TO PRINTER...", enter **3** to specify that you want 3 copies of each letter printed, for a total of 12 letters.

Printing Labels from the Same Data File

If you output the primary file above through FORMAT, four copies of the form letter will be printed. You can use the primary file below with the same data file to print a label for each letter's mailing envelope. You must have appropriate labels loaded in your printer, of course, on which to print the labels.

```
.OP CUSTOMER LIST,S6,D2
.* Set the page length for each label's
.* depth, in this case, 6 lines from
.* label top to label top.
.PL 6
.* This is a sample of what two output
.* labels will look like:
.*
.* MANAGER'S NAME 1 LABEL START
.* 2
.* STORE NAME 3
.* STREET ADDRESS 4
.* CITY, ST ZIP 5
.* 6
.* MANAGER'S NAME 1 LABEL START
.* 2
.* STORE NAME 3
.* STREET ADDRESS 4
.* CITY, ST ZIP 5
.* 6
.*
.* Set top and bottom margin spacing to zero
.M1 0
.M2 0
.M3 0
.M4 0
.* Read the manager's name
.* and output 1 line space
.RB
.SP
.* Read the store name and address
.RB
.* Make sure that the data blocks
.* are read accurately
```

```
.NG
.* Line 6 will be skipped when FORMAT
.* starts a new 6-line "page" for
.* the next label.
```

Printing the primary file above via FORMAT will produce a label for each of the four form letters. The group terminators and the **.ng** command protect you from printing several inaccurate labels.

PIE Writer's Mail Merge capability, of course, has several applications that have nothing to do with mailing. A standard weekly report that accesses data blocks from any of several data files is just one useful application.

The operator can be queried for the title of the file to be merged simply by inserting an **.op** command in the primary file without naming the data file to be opened. In this way, a source file could be developed that gets a data file's name from the user (via the keyboard) to determine which data file to open.

TERMINAL COMMUNICATION

Simply stated, FORMAT obtains the information it processes in one of two ways. The usual sequence is for FORMAT to process the information in the text buffer. Sometimes this requires FORMAT to find a file stored on diskette or to chain to another file by loading that file into the text buffer. In these cases, once formatting begins, the operator is no longer involved in the process.

The second option allows the operator to provide some information from the keyboard that will be processed either into the output document (the date, perhaps) or to alter the commands (for instance, by selecting single or double spacing). Obtaining input from the operator in this manner is referred to as "terminal communication" because FORMAT gets the data it processes from the data entry terminal (the keyboard) rather than the text buffer or a source file.

Typically, there are two specific situations in which it may be desirable to direct FORMAT to display a prompt asking the operator to enter data while FORMAT is processing a file.

Physical Printer Adjustment

Many printers require that a daisy wheel, thimble, or selectric ball be changed to use a different type font for emphasis. In this instance, the escape character and plus sign (]+) are used. This command stops printing immediately and waits for a key to be pressed before continuing. By preceding this command with a **.tm** command, you can create a video prompt string that tells the operator exactly what adjustment must be made to the printer. For instance,

```
.TM "CHANGE TO WHEEL 5 AND PRESS ANY KEY"
```

or

```
.TM "CHANGE TO RED RIBBON AND PRESS ANY KEY"
```

notifies the operator to make the necessary change when the]+ is encountered by FORMAT. FORMAT waits, then continues when any key is pressed.

If the FORMAT output is directed to a disk file the action of the]+ command is deferred until the file is printed. If you are "spooling" your output for later printing, FORMAT replaces the]+ with a CTRL-C in the output. It is the responsibility of the spooling software to recognize this situation and obtain operator actions.

Altering FORMAT Command Variables

The second application for terminal communication allows you both to set formatting instructions and to add lines of text interactively from the keyboard while you are formatting.

Suppose that you wish to create a standard "header" file that allows you to set up a variety of documents for printing. Sometimes you will need a double-spaced draft for easy editing before printing a final single- spaced document. The following command sequence takes these needs into account:

```
.* Set line space command
.LS]
.* Enter line spacing
.GL Press 1 for final, 2 for draft
```

The] character connects the **.ls** to the **.gl**. It tells FORMAT that the value for the **.ls** command will come from the **.gl**, which will ask the user to input the value at the keyboard. The **.gl** command tells FORMAT to stop processing, provides FORMAT with a video screen prompt, and instructs FORMAT to use the value entered at the keyboard in response to the prompt as the **.ls** variable. According to the prompt above, if the operator enters a **1**, a single-spaced final document will be output, and if a **2** is entered, a double-spaced draft will be produced.

When printing an often-used form letter on a regular basis, you may need to change only a single piece of information, such as the date, each time the form letter is prepared:

```
.* Set up date entry
.TL''']
.GL Enter today's date (mon dy, yyyy)
```

The above would instruct the operator to place the proper date in the form letter with a simple prompt.

Altering the salutation for each letter could be arranged in the same way:

```
.* Set up salutation entry
Dear]
.GL Enter name and comma
.SP
Text of letter
```

SHELL FILE STRUCTURES

Shell file structures allow you to perform frequently used functions without risking errors due to tedious typing. A shell file contains an exact copy of a sequence of commands that you normally would enter from the keyboard.

To execute a shell file:

1. *Type* **%E filename**
2. *Press* **RETURN**

Every one of the commands in the file will be executed before control is returned to the CP (unless one of the commands requires that the operator enter variables from the keyboard).

This feature makes tasks such as copying and backing up files a simple process. On your master diskette, a shell file called BACKUP has been provided that will do the following:

1. *P-*
2. *CALL $830 (restore working drive)*
3. *"BACKING UP FILE: "?"*
4. *%D ?.BAK*
5. *%R ?,?.BAK*
6. *S ?*
7. *P+*

The commands are interpreted as follows:

1. *Suppress printing of normal CP prompts*
2. *Restore current data drive, slot, volume*
3. *Print prompt "BACKING UP FILE: FILENAME"*
4. *Delete the file "filename.BAK"*
5. *Rename "filename,filename.BAK"*

6. *Save binary filename*

7. *Enable printing of normal CP prompts*

Shell File Commands

The **P-** command suppresses and the **P+** command enables the normal CP prompts, and they are valid CP commands in themselves.

When a double quote (") is placed in a shell file, all the characters that follow prior to a second double quote are displayed on the video screen.

To print a blank line on the screen, enter two double quotes ("").

A special feature allows you to change the name of your default file.When an "I" is entered at the keyboard, the next line input becomes the default filename. When this is executed within a shell file, the current file can be manipulated and then output with the new filename obtained from the keyboard.

SHIFT KEY MODIFICATION

One of the unfortunate design characteristics of the Apple II, from a word processing point of view, is that the **SHIFT** key does not give you access to uppercase letters as it does on a standard typewriter. PIE uses the ➡ key as a shift key instead. However, you can make a modification to your Apple that allows the **SHIFT** key to give you access to uppercase when you are in the lowercase mode.

The modification described here is for Revision 7 or more recent Apples. It is quite possible to make this modification without soldering. Before following the steps below, however, note that ANY PERMANENT MODIFICATION OF THE APPLE MAY VOID YOUR WARRANTY.

1. *Turn the power off and remove the cover.*

2. *Take a 10-inch piece of light gauge, unstranded wire, strip it at both ends, and bend one end into a hook about a half-inch long.*

3. *Locate the keyboard encoder board to the right underneath the front edge of the cover opening (more or less underneath the keyboard). There are 25 stiff wires connecting the keyboard encoder to the keyboard itself.*

4. *Facing your Apple with the keyboard in front of you and looking down at the encoder card, locate the second wire from the right, the first being the wire on the extreme right.*

5. *Snake the hooked end of the unstranded wire around this second wire, which is connected to the* **SHIFT** *key. With a piece of (preferably electrical) tape, close the hook so that there is a tight loop around the second wire. The idea is to ensure that the wire does not make contact with any of the other wires on the encoder board.*

6. *Having done this, attach the other end of the wire to pin 4 (pin 1 is in the lower right-hand corner) of the game controller card. {If you have one, you could instead attach it to the "lowercase pad" on the DoubleVision 80-column card.}*

7. *You now must run PIE CONFIGURE (and FORMAT CONFIGURE if you use the* **.gl** *command) to modify PIE Writer so that it will recognize the* **SHIFT** *key modification you have just made.*

This completes the installation. The **SHIFT** key now will produce uppercase letters when held down with a letter key when in the lowercase mode. In the uppercase mode, the keyboard functions normally, meaning that a **SHIFT-N**, for example, generates the up arrow.

To enter the standard letter-shift characters from lowercase mode, type the following:

To get:	You type:
@	⇨ (CASE) * or :
]	⇨ (CASE) > or .
^	⇨ (CASE) " or **2**

TELECOMMUNICATION

The Apple II and PIE Writer provide a very potent telecommunication system.

It is possible to "hookup" with a distant computer via a telephone line and modem. With such a connection established, it is not difficult to obtain the information in another computer's text buffer or to send the contents of your text buffer to another computer. The power of word processing and the immediacy of telecommunication constitute a very desirable information system.

Two Apple computers running PIE Writer can exchange files of information. The process is not complex, but requires one or more pieces of specialized equipment:

1. *A RS-232 serial controller card with an acoustic coupler. Two tested serial cards are the Apple Communications Card and the CCS7710A Serial. Novation's CAT is a tested acoustic coupler.*

or

2. *A Hayes Micromodem II.*

The following has been tested only with the 40-column version of PIE Writer.

Transmitting Text with a Communications Card

To transmit text via a card, enter the Command Processor, and type:

1. < **#slot** *(#slot is the slot the Comm Card is in)*
2. *Press RETURN*

Although nothing appears to happen, this command sets up the "firmware" of the serial card. Either dial the telephone to the distant computer manually and listen for the "carrier whistle," or if your card supports auto-dial, use the appropriate commands to establish the connection.

Now with the terminal connected and the COMMAND?: prompt displayed, you may enter control codes that signal commands to the card's firmware. (The card should support a "terminal mode," as the Hayes card does, for example.) At this point, PIE Writer is cut off temporarily from the computer's activities. It remains in memory but no longer participates in the transmission of data.

Use the terminal mode to enter whatever commands the receiving terminal requires to catch the information being transmitted.

Now exit the card's terminal mode to return the COMMAND?: prompt. At this point, the Command Processor is active and the Communication Card is inactive. You remain "connected" with the other computer, but for now, your end of the connection is "quiet," while the other end expectantly awaits the transmission in a "stream" of characters.

Load the file you wish to transmit from the diskette into the text buffer, and enter the edit mode by typing:

1. **E$**
2. *Press* **RETURN**

This takes you to the end of the file.

Now we will pick a character, CTRL-B, for instance, and type it in as the last character in the file using the **CTRL-SHIFT-M** (QUOTE) command, followed by the CTRL-B. This CTRL-B character acts as a marker. Note in the ASCII Reference Chart on p. 94 that the decimal value of CTRL-B is 130. Press **CTRL-SHIFT-P** to return to the CP.

To send the entire contents of the text buffer to the other computer, type the following:

1. **@130>#slot**

2. Press **RETURN**

There will be no indication that information is being transmitted except that the cursor will flash at ragged intervals. The characters will be output to the other computer as if it were a local printer or your video screen.The printer card controls the speed, usually about 30 characters a second.

When the transmission is completed, the "COMMAND?:" prompt reappears. Reenter the terminal mode. Do whatever is necessary to verify the text transmitted to the receiving computer.

Receiving a File with the Communication Card

It is just as likely that you will want to receive the output of another computer in your PIE text buffer.

In the previous example:

>#slot

sent all the characters out the telecommunications port. To receive characters, you must tell the CP that you want to bring all characters in FROM a slot. This is done as follows:

1. Type **<#slot**

2. Press **RETURN**

If there is no card in the slot, the Apple II will hang, in which case nothing but a **RESET** will return the CP prompt.

It is necessary for the text material being sent from the distant computer to end with a known marker character that will not appear in the regular text.

If the transmitting computer is capable of sending, for instance, a CTRL-B, enter:

@130 <#slot

Again, 130 is the decimal ASCII representation of CTRL-B. This command allows you to receive every character that comes through "#slot" until the character CTRL-B appears, at which point transmission ends. If the CTRL-B is never sent, the Command Processor will wait forever in expectation of that character.If this happens, press **RESET**, and request that file be re-transmitted.

When the transmission is complete, you can edit the transmitted file or save it to diskette. You may disconnect your modem or sign off.

Apple to Apple Communication

The best of all telecommunication environments is when working with two Apple II computers equipped with PIE Writer. This allows direct transmission of data from one word processor to another.

It is not possible to see the data being sent or received during transmission. PIE Writer was not designed as a telecommunications device, but its flexibility makes this process possible.

Also, if during a transmission to a distant computer it takes too long to process a line (more than one "character-time" after it is transmitted), there may be a loss of characters at the other end. PIE Writer cannot send a delay after every carriage return, so it is possible for the receiving computer to lose some of the transmission because it is busy processing its input.

ADDRESS TABLES

Address tables are useful to programmers who want to write utility programs that extend PIE Writer's capabilities even further.There are several types of addresses that may be of interest:

1. *Page zero pointers, entry points, and special locations.*
2. *The PIE and FORMAT tables normally altered via PIE Configure and FORMAT Configure.*
3. *The keyboard "maps" that tell PIE Writer which function to perform.You may customize the keyboard according to your own needs following the procedure described below under "Command Table Modification."*

All of the above types of addresses follow.

Page Zero Buffer Pointers

$06	Beginning of text ($4000 for 40-column version; $4C00 for 80-column)
$08	End of text + 1 (points to last byte, which contains a 0)
$0A	End of text buffer, used only by PIE (default = $95FF)

Important PIE Addresses

$803	Cold start address (can **CALL 2051** from BASIC)
$806	Warm start address (can **CALL 2054** from BASIC)
$809	Restart address (for Apple II's without autostart ROM)
$80C	Hook for intercepting commands to CP AUX program
$815	High bit set on output to printer if = 1 (default)
$816	Default buffer begin address ($4000 for 40-column; $4C00 for 80-column)
$818	Default buffer end address ($95FF)
$81F	Convert break line token (default = CTRL-Z with high bit set)
$820	Printer slot (default = 1)
$821	Line feed after return if = 1 (default = 0)
$822	Printer initialization string (7 characters maximum, plus 0 terminator)
$82A	Program disk volume (default = 0)
$82C	Program disk drive (default = 1)
$82E	Program disk slot (default = 6)
$1290	High bit set in text buffer if = 1 (default)
$1291	Initial bell column (default = 38 for 40-column; 78 for 80-column)
$1292	Initial tab increment (default = 8)
$1293	PPWrap initialization (default = 0 [off]; 1 = PPWrap 2 = Indent)
$1294	Initial lowercase input if = 1 (default)
$1295	Output lowercase to CRT if = 0 (default = 1)
$1296	Address of **SHIFT** key mod input port (default = none)
$1298	Mask for **SHIFT** key bit
$1299	Mask for inverting **SHIFT** key bit
$129A	Value OR'ed with control characters to display (=$40 for 40-column)

$12A0-$12DF	Command table 1 (commands with no arguments or non-null arguments)
$12E2-$1321	Command table 2 (commands with null arguments and cursor-defined arguments)
$1324-$1333	Character translation table for ➡ (CASE) key
$3300-$39FF	Screen buffer (available to user for executing AUX programs from the Command Processor)

Important FORMAT Addresses

$803	Cold start address (can **CALL 2051** from BASIC)
$806	Warm start address (can **CALL 2054** from BASIC)
$809	Restart address (for Apple II's without autostart ROM)
$811	Number of lines to scroll before pausing (default = 24)
$813	Printer address (default = $C100)
$816	Output lowercase to CRT if = 1 (default = 0)
$817	Page stop always enabled if = 1 (default = 0)
$818	Default buffer begin address ($4000 for 40-column; $4C00 for 80-column)
$81A	Underline mode (0 = none; 1 = BS; 2 = SS; 3 = Overlay; default = 1)
$81B	Initial escape character (default = ']')
$81D	Initial data block terminating character (default = '<')
$81E	High bit set on output to printer if = 1 (default)
$81F	Line feed output after return if = 1 (default = 0)
$820	Initial right adjust if = 1 (default = 0)
$821	Initial fill mode if = 1 (default)
$822	Initial line spacing (default = 1)
$823	Initial margin 1 value (default = 2)
$824	Initial margin 2 value (default = 2)
$825	Initial margin 3 value (default = 1)
$826	Initial margin 4 value (default = 3)
$827	Initial paragraph indent (default = 5)
$828	Initial paragraph indent sign ('+', '-', or ' '; default = '+')
$829	Initial paragraph spacing (default = 1)
$82A	Initial paragraph need value (default = 2)
$82B	Initial page length (default = 66)
$82C	Initial line length (default = 65)
$82D	Initial page offset (default = 0)
$82E	Initial offset character (default = ' ')

$830	Underline start sequence (3 characters maximum, plus 0 terminator)
$834	Underline stop sequence (3 characters maximum, plus 0 terminator)
$838	Boldface start sequence (3 characters maximum, plus 0 terminator)
$83C	Boldface stop sequence (3 characters maximum, plus 0 terminator)
$840	Printer initialization string (7 characters maximum, plus 0 terminator)
$848	Printer type (1 = Diablo; 2 = Qume; 3 = NEC; 4 = Epson; 5 = Centronics; 6 = other; default = 6)
$849	Width of character in 1 120's for daisy-wheel printers (= 12 for 10 characters-per-inch)
$850	Initial lowercase input (**.gl**) if = 1 (default)
$851	Boldface mode (0 = none; 1 = BS; 2 = SS; default = 1)
$852	Number of strikes per character for BS boldface (default = 3)
$853	Stop character on output to disk (default = CTRL-C
$854	Line feed after carriage return on output to disk if = 1 (default = 0)
$855	Address of **SHIFT** key mod input port (default = none)
$857	Mask for **SHIFT** key bit
$858	Mask for inverting **SHIFT** key bit
$859	Value OR'ed with control characters to display (= $40 for 40-column)
$85A	Program disk volume (default = 0)
$85C	Program disk drive (default = 1)
$85E	Program disk slot (default = 6)

Addresses Common to PIE and FORMAT

$3F40	Remembered filename, terminated by **RETURN**
$3F76	Remembered buffer end address
$3F80	Remembered tab settings
$4000	Beginning of text buffer (40-column)
$4C00	Beginning of text buffer (80-column)

Locations $F8-$FF and $300-$3CF are reserved for the user (such as for a printer driver). All other memory below the top of the text buffer (for instance, $95FF) is reserved for use by PIE and FORMAT, except for the AUX program buffer at $3300 as described above.

FORMAT Limits

- *Maximum page length: 250 lines*
- *Maximum page number: 9999*
- *Maximum output line length: 132 characters*

Command Table Modification

The keyboard layout for PIE is all table-driven, and it is possible to reconfigure the keyboard. There are two command tables indicated above: one for commands with no arguments or with a non-null argument, and the second for commands with a null argument.

Each table has 32 addresses, one for each control character (0-$1F).To rearrange the keyboard, just swap addresses corresponding to the appropriate keys. If the addresses in both tables for the same key are the same, then both tables should be modified together.

Custom Program Area

The area of the computer's memory that is dedicated to the screen display in edit mode is free when you are in the Command Processor. When a "%" command is encountered, the PIE AUX file is loaded into this area ($3300).

If you want to write a utility or set of utilities using the address tables above, go right ahead. All code should reenter PIE Writer through the DOS $3DO softstart vector. Use the **BLOAD** and **CALL** or **BRUN** commands to execute your utility.

Index

See also file chaining pg. 127

The type for this book was set using PIE Writer to send text to Hayden's in-house typesetting system. Finished books were off press a week and a half after the transmission began. The majority of typesetting commands were imbedded using PIE Writer prior to transmission. A small machine-language subroutine, invoked from PIE Writer's command processor with a CALL statement, managed the interface between the Apple and a Bedford customized DEC PDP-11.